The NOMAD System: Expectation-Based Detection and Correction of Errors during Understanding of Syntactically and Semantically Ill-Formed Text[1]

Richard H. Granger

Artificial Intelligence Project, Computer Science Department
and Cognitive Sciences Program
University of California
Irvine, CA 92717

Most large text-understanding systems have been designed under the assumption that the input text will be in reasonably "neat" form (for example, newspaper stories and other edited texts). However, a great deal of natural language text (for example, memos, messages, rough drafts, conversation transcripts, etc.) have features that differ significantly from "neat" texts, posing special problems for readers, such as misspelled words, missing words, poor syntactic construction, unclear or ambiguous interpretation, missing crucial punctuation, etc. Our solution to these problems is to make use of *expectations,* based both on knowledge of surface English and on world knowledge of the situation being described. These syntactic and semantic expectations can be used to figure out unknown words from context, constrain the possible word senses of words with multiple meanings (ambiguity), fill in missing words (ellipsis), and resolve referents (anaphora). This method of using expectations to aid the understanding of "scruffy" texts has been incorporated into a working computer program called NOMAD, which understands scruffy texts in the domain of Navy ship-to-shore messages.

1. Introduction

The NOMAD system takes unedited English input in a constrained domain, and works interactively with the user to encode the message into database-readable form. The unedited texts in this domain are Naval ship-to-shore messages, written in 'telegraphic' English, often leaving out nouns and verbs, crucial punctuation (such as periods), and making use of ad hoc abbreviations of words. In addition to these problems of *surface-text* processing, these texts can contain problems of interpretation – that is, which of several objects is being referred to, or which possible goal inference is implied. These *semantic* processing problems are not easily detectable or solvable based on the surface text alone but rather require a data base of knowledge about the domain of discourse, in this case ship movements.

Here are examples of each of these two types of problems. First, one with a number of surface-text errors:

(1) 'Locked on open fired destroyed'

Example (1) is missing crucial punctuation (no boundaries separating the three clauses from each other), is missing subjects and objects for all three verb phrases, and has a tense mismatch in the middle phrase ('open fired'). (This is an actual message in the corpus provided to us by the Navy, not a constructed example.) NOMAD's output from this example is:

> We aimed at an unknown object.
> We fired at the object.
> The object was destroyed.

A second message has, in addition to some surface problems, a goal-based interpretation problem:

[1] This research was supported in part by the Naval Ocean Systems Center under contracts N-00123-81-C-1078 and N66001-83-C-0255, and by the National Science Foundation under grant IST-81-20685.

0362-613X/83/030188-09$03.00

(2) 'Returned bombs to Kashin.'

In addition to the surface problem of a missing subject, this example is apparently missing mention of some previous event, implied by the use of 'returned'; and it describes an ambiguous event, that is, either the peaceable delivery of bombs to the Kashin ship (a type of enemy ship) or a battle action of firing bombs in retaliation. Since the input is ambiguous without the previous message, NOMAD returns a number of alternative possible outputs to the user, marking one as "preferred":

(Preferred Interpretation):
We fired some bombs at a Kashin ship.
(Inferred):
The Kashin ship fired at us previously.

(Alternate Interpretation):
We delivered some bombs to a Kashin ship.
(Inferred):
The Kashin ship had delivered some bombs to us previously.

NOMAD is interactive: it produces multiple interpretations when necessary, and lets the message-sender choose among these alternatives. A typical scenario is:

- the user (message-sender) will enter a 'telegraphic' message;
- NOMAD will produce two different possible interpretations of the message in corrected English, and present them to the user;
- the user will then choose one of the interpretations; and
- a database-readable version of the correctly-interpreted message is then forwarded from the ship to a central data base.

Many of the approaches to understanding ill-formed input focus on syntactic errors separately from semantic errors (for example, Hayes and Mouradian 1981 and Kwasny and Sondheimer 1981). Both of these efforts essentially attempt to increase the flexibility of an ATN syntactic parser: the first by using 'parse suspension and continuation', relaxing constraints on consistency and permitting matches out of their correct order, and the second by relaxing the constraints required to traverse an ATN arc, and then providing 'deviance notes' specifying the differences between what was expected and what was actually seen. These efforts attempt to correct the surface form of the input, that is, to perform a transformation from an ill-formed English text to a well-formed English text. Their goals are not to produce a meaning representation of the input, and hence cannot be said to 'understand' the input. This also leads to the inability of these systems to generate alternative interpretations of text; once these systems have guessed at a parse, they cannot back up and re-parse in response to information from a user.

The approach taken by Hayes and Carbonell (1981) is closer to that described in this paper, in that they do build meaning representations. However, there are still shortcomings; in particular, their systems cannot understand texts in which a missing or unknown word is the one that would have built the main semantic case frame. As will be seen below, NOMAD builds on the FOUL-UP system (Granger 1977) to handle such cases (which are frequent in our domain). Furthermore, like the systems described above, their systems cannot re-interpret a text when its initial interpretation turns out to be incorrect.

We propose an integrated system of syntactic and semantic processing, in which world knowledge and syntactic knowledge are both applied during text processing to provide a number of possible interpretations of a text. Our focus is on interpretations: the goal of the system is to give rise to an unambiguous meaning representation. If surface-text problems occur during processing but an unambiguous interpretation can be provided and confirmed by the user, then the surface-text problems are ignored. It is only when interpretation problems arise that any noted surface-text problems will be consulted to see if they might have been the source of the interpretation problem. That is, we are attempting to attack the overall problem of processing text, of which the processing of ill-formed text is a necessary subpart. Our approach implies that the processing of ill-formed text 'falls out' of normal text processing, via the application of generalized error-correction processes that operate equally on syntax, semantics, and pragmatics, and are not designed specifically for the processing of ill-formed surface text.

NOMAD builds on previous work on conceptual analysis (Riesbeck and Schank 1976, Birnbaum and Selfridge 1979), and on error detection and correction during conceptual analysis (Granger 1977, 1980, 1982a). Selfridge and Engelberg (1984), Lebowitz (1984), and Dyer (1983) have also recently taken approaches that are similar to the one proposed here, attempting to fully exploit the power of integrated understanding. NOMAD incorporates and integrates error detection and correction algorithms based on both syntactic and pragmatic error types, and is therefore capable of correctly processing a wide range of ill-formed texts within the knowledge domain of Navy messages. NOMAD has actually been installed and is being used for message processing by the Naval Ocean Systems Center (NOSC) at San Diego.

2. Background: Tolerant Text Processing

2.1. FOUL-UP figured out unknown words from context

The FOUL-UP program (Figuring Out Unknown Lexemes in the Understanding Process; Granger 1977) was the first program that could figure out meanings of unknown words encountered during text understanding. FOUL-UP was an attempt to model the corresponding human ability commonly known at "figuring out a word from context". FOUL-UP worked with the SAM system (Cullingford 1977), using the expectations generated by scripts (Schank and Abelson 1977) to restrict the possible meanings of a word, based on what object or action would have occurred in that position according to the script for the story.

For instance, consider the following excerpt from a newspaper report of a car accident:

(1) Friday, a car swerved off Route 69. The vehicle struck an embankment.

The word "embankment" was unknown to the SAM system, but it had encoded predictions about certain attributes of the expected conceptual object of the PROPEL action (the object that the vehicle struck); namely, that it would be a physical object, and would function as an "obstruction" in the vehicle-accident script. (In addition, the conceptual analyzer (ELI – Riesbeck and Schank 1976) had the expectation that the word in that sentence position would be a noun.)

Hence, when the unknown word was encountered, FOUL-UP would make use of those expected attributes to construct a memory entry for the word "embankment", indicating that it was a noun, a physical object, and an "obstruction" in vehicle-accident situations. It would then create a dictionary definition that the system would use from then on whenever the word was encountered in this context.

2.2. Syntactic (surface) and semantic (interpreation) text errors

But even if the SAM system had known the word "embankment", it would not have been able to handle a less edited version of the story, such as this 'telegraphic' message, which might have been sent in by an on-the-scene reporter:

(2) Vehcle acc Rt69; car strck embankment; drivr dead one psngr inj; ser dmg to car full rpt frthcmng.

While human readers would have little difficulty understanding this text, no existing computer programs could do so.

The scope of this problem is wide; examples of texts that present "scruffy" difficulties to readers are completely unedited texts, such as messages composed in a hurry, with little or no re-writing, rough drafts, memos, transcripts of conversations, etc. Such texts may contain these problems, among others: missing words, ad hoc abbreviations of words, poor syntax, confusing order of presentation of ideas, misspellings, lack of punctuation. Even edited texts such as newspaper stories often contain misspellings, words unknown to the reader, and ambiguities; and even apparently very simple texts may contain alternative possible interpretations, which can cause a reader to construct erroneous initial inferences that must later be corrected (see Granger 1980, 1981a, 1981b).

The following sections describe the NOMAD system, which incorporates FOUL-UP's abilities as well as significantly extended abilities to use syntactic and semantic expectations to resolve these difficulties, in the domain of Navy messages. NOMAD's processing is divided into two major categories:

(1) *blame assignment,* that is, the detection of an error and the attribution of that error to some source; and
(2) *error correction,* the remedy for the source of the error.

3. How NOMAD Recognizes and Corrects Errors

3.1. Introduction

NOMAD incorporates ideas from, and builds on, earlier work on conceptual analysis (for example, Reisbeck and Schank 1976, Birnbaum and Selfridge 1979), situation and intention inference (for example, Cullingford 1977, Wilensky 1978), and English generation (for example, Goldman 1973, McGuire 1980). What differentiates NOMAD significantly from its predecessors are its error recognition and error correction abilities, which enable it to read texts more complex than those that can be handled by other text understanding systems.

NOMAD operates by attempting to process texts left to right, with each word capable of suggesting new expectations (for example, a verb will follow, the previous noun group should serve as actor of the current act, etc.), and applying those suggested expectations to new inputs. When expectations are met, they result in additions to the ongoing meaning representation of the text; when they are not met, they result in 'surface-text alerts', which are collected for potential later corrective processing.

There are two types of 'errors' in NOMAD: surface-text errors and interpretation errors. Surface-text errors are potential problems that can be readily detected at surface-text processing time, including, for example, unknown words and any surface expectation violations, whether syntactic or semantic. For instance, a syntactic expectation failure such as 'no noun

group appearing where one was expected' is a surface alert, but so is a semantic/pragmatic expectation failure such as 'target noun group was expected to describe an animate actor, but described an inanimate object instead'. Each of these is equally an expectation failure, and no difference need be drawn at this stage of processing between syntactic or semantic types. It will be seen later that, depending on the type of surface alert, different suggestions will be made as to where to look to 'assign blame' for the problem, and how to attempt to correct it.

Interpretation failures, on the other hand, are defined as those that cannot be easily ascribable to the failure of some particular pre-defined surface expectation; these arise after some conceptual analysis has been successfully performed and the resulting representation fails to match pragmatic checks such as goal-based or script-based knowledge of the situation being described.

Following is a list of nine categories of problems we have identified that occur often in scruffy unedited texts, five surface-text problems and four interpretation problems. Each problem is illustrated by a brief example from the domain of Navy messages. It will be seen that these errors often occur in pairs, with surface-text problems sometimes giving rise to interpretation problems. Note that while these problems are often referred to in this paper as 'errors' in fact some are not actual 'errors', strictly speaking, but are rather *potential* problem indicators that NOMAD recognizes, which may give rise to subsequent interpretation problems.

Surface-text problems

1. Unknown words.
   Enemy 'scudded' bombs at us. – the verb is unknown to the system.
2. Missing subject, object, etc. of sentences.
   Sighted enemy ship. Fired. – the actor who fired is not explicitly stated.
3. Missing sentence and clause boundaries.
   Locked on opened fire. – two actions, aiming and firing.
4. Ambiguous word usage.
   Returned bombs to Kashin. – "returned" in the sense of retaliation after a previous attack, or "returned" in the sense of "peaceably delivered to"?
5. Lack of tense agreement.
   Open fired. – the intended tense of 'open' is transferred to 'fire'.

Interpretation problems

1. Causality violation.
   Ship sighted overhead. – ships can't fly; probable message-sending error.
2. Goal violation.
   Returned bombs to Kashin. – one of two ambiguous interpretations of 'returned' (peaceably delivered) gives rise to apparent goal violation (delivering weapons to enemy).
3. User confirmation failure.
   NOMAD's failure is not confirmed by user. (Note that this is considered by NOMAD to be an interpretation problem even thought it may be due to the user's idiosyncrasies, as opposed to violation of some known semantic rule – the effect is the same.)
4. Object or event referenced out of known event sequence.
   Midway lost contact on Kashin. – no previous contact mentioned; this often arises when typical known situations are mentioned in other than stereotypical (scripty) order.

When these problems arise in a message, NOMAD must first recognize what the problem(s) is(are) (which is often difficult to do), and then attempt to correct the error(s). The following section outlines the overall processing algorithms NOMAD uses to process these errors.

3.2. NOMAD's error-detection algorithm

NOMAD's algorithm for detection and solution of errors follows a four-step process:

1. Set 'alert' flags wherever potential surface-text problems are detected.
2. Do only partial processing of surface text if necessary due to missing or ambiguous information (that is, do as much normal processing as possible in the face of missing information).
3. Check for interpretation problems (causal, goal, sequencing (script), or user confirmation errors) after surface sentence processing.
4. Try solutions based on surface 'alert' flag categories.

To illustrate this process, consider an ambiguous text, 'contact gained on kashin'. During the processing of this text, some surface-text alerts arise (for example, 'contact' can be either a noun or a verb' if it's a verb, then there's either a missing subject or an expected passive subject coming, etc.), and an interpretation ambiguity: the text can be interpreted as meaning either

(a) We established visual or radar contact with a kashin ship.
(b) Our contact (that is, a ship in contact with us) increased its speed in a chase after a kashin ship.

In the case of 'contact gained on kashin', NOMAD's blame assignment algorithm moves through the above steps as follows:

1. **(a)** Set both 'ambiguous-word-sense' and 'ambiguous-part-of-speech' alerts for the word 'contact': it might be either a noun (that is, the ship that is currently our contact) or a verb (to establish radar or visual contact).

(b) Set 'ambiguous-word-sense' alert for word 'gained': it might mean either 'established' as in 'gained (established) radar contact', or 'advanced' as in 'gained (advanced) on enemy during chase'.

2. Product alternate interpretations based on alternate assumptions about word senses: 'established radar or visual contact with kashin', and 'our contact ship advanced on kashin'.
3. **(a)** Look for possible causality or goal violations: none found.
   (b) Ask user for confirmation: user confirms one interpretation but not the other.
4. Solution: Select interpretation confirmed by user.

Consider another example, 'Returned bombs to Kashin'. As noted above, one of two ambiguous interpretations of 'returned' in this text (that is, the '(peaceably) delivered' interpretation) gives rise to an apparent goal violation (delivering weapons to enemy).

In the case of 'returned bombs to kashin', the blame assignment algorithm acts as follows:

1. **(a)** Set 'ambiguous-word-sense' alert for the word 'returned': it might have either of two categories of meaning, corresponding to 're-do a previously-done *action* (as in 'return the favor', 'return a transmission') or 're-deliver a previously-delivered *object* (as in 'return a (borrowed) book').
   (b) Set 'ambiguous-word-sense' alert for word 'bombs': it might mean either the verb 'to bomb', present tense, or the plural noun. The former interpretation (that bomb is a verb) also gives rise to a 'missing-clause-boundary' surface alert, since then the 'returned' and 'bombs' verbs would be next to each other.
2. Produce alternate interpretations based on alternate assumptions about word senses: 'Delivered object (bombs) to kashin' (after they had delivered some to us) or 'fired on kashin' (after they had fired on us). (The error-ridden alternate interpretations that arise from the verb sense of 'bombs' are also generated.)
3. **(a)** Look for possible causality or goal violations: With the 'delivery' interpretation, a potential violation of one of NOMAD's known goals is found: Actors of class (enemies) transferring possession of objects of class (weapons) to recipients of class (friends), and vice versa.
   (b) Order the interpretations in order of preference, based on both surface-class and interpretation-class errors; the goal-violation case above is not preferred, and the 'bombs-as-verb' case is not preferred, while the 'firing back at kashin' interpretation is preferred.
   (c) Present preferred interpretation to user; confirmed. (If this had failed, then unpreferred interpretations would have been presented.)
4. Solution: Select confirmed interpretation.

4. Blame Assignment in NOMAD

As evidenced in the above examples, there is no simple relationship between types of errors in the interpretation of the input, and possible solutions to those errors. This is primarily because the *source* of an interpretation error is difficult to identify. In general, interpretation problems can arise from any of a number of surface-text problems, including:

1. words with multiple word senses
   Returned bombs to Kashin. – see above discussion;
2. missing clause boundaries
   Challenged ship refused to heave to. – can be interpreted in any of the following ways: (a) We challenged a ship. They refused to heave to. (b) We challenged a ship. We refused to heave to. (c) The challenged ship refused to heave to.
3. elliptical or telegraphic sentence construction
   Contact gained on Kashin. – can be interpreted as: (a) We established visual or radar contact with a kashin ship. (b) Our contact (that is, a ship in contact with us) increased its speed in a chase after a kashin ship).

As mentioned earlier, NOMAD's goal is to produce correct, unambiguous interpretations of input texts. Its ability to handle ill-formed surface text arises from a need to be able to find surface-text problems that give rise to interpretation problems; it attends to surface-text problems not because they are useful in their own right but only because they may be useful later in solving an interpretation problem. NOMAD collects both surface-text problems and interpretation problems as it processes a text, and for each interpretation problem, it attempts to find a corresponding surface problem that gave rise to it. Once it has an interpretation problem – surface problem pair, it suggests a solution for the overall problem based on the characteristics of both the surface problem and the interpretation problem. In cases where only a surface problem exists and no interpretation problem has arisen, the surface problem is simply ignored as being irrelevant to the true understanding goal of producing a correct, unambiguous interpretation. In cases where an interpretation problem exists but no surface-text problem can be linked to it, NOMAD suggests possible solutions to the interpretation problem that do not depend on surface problems.

The 'blame assignment chart' below illustrates some of NOMAD's heuristics for finding surface-text alerts that might correspond to a given interpretation problem.

NOMAD's blame assignment algorithm is at the center of its ability to handle syntactically and semantically ill-formed text. Blame assignment in NOMAD is capable of dealing with problems at both the surface-text level and the interpretation level, especially where interpretation problems arise indirectly from surface-

INTERPRETATION PROBLEM	SUGGESTED SURFACE-TEXT ALERT	SUGGESTED POTENTIAL SOLUTIONS
Only partial representation constructed	Unknown word	FOUL-UP: Expectations and Act-Preference
Causality violation, Goal violation, User confirmation failure	Word with multiple word senses	Try alternate word sense
	Expectation failures: –Syntactic (word) –Semantic –Boundary (phrase)	Try inferring clause break
Actor or object reference out of sequence	(No surface alert)	Try situation-frame inference
Event referenced out of sequence	(No surface alert)	

Blame Assignment Chart

level decisions; in general, there is no simple relationship among surface-text problems, interpretation problems, and potential solutions for these problems.

4.1. Recognizing and correcting surface errors

For each of the five categories of surface problems handled by the system, NOMAD's method of recognizing and correcting the problem is briefly described here, along with actual English input and output from NOMAD.

1. INPUT:
ENEMY SCUDDED BOMBS AT US.

Problem: Unknown word. The unknown word "scudded" is trivial to recognize as being unknown, since it is the only word without a dictionary entry. Once it has been recognized, NOMAD checks it to see if it could be (a) a misspelling, (b) an abbreviation, or (c) a regular verb-tense of some known word.

Solution: Use expectations to figure out word meaning from context. When the spelling checkers fail, a FOUL-UP mechanism is called that uses syntactic expectation (and some morphological analysis) to infer that 'scudded' is probably a verb, and then uses pragmatic knowledge of what actions can be done by an 'enemy' ACTOR, to a 'weapon' OBJECT, direct TO us. At this point, NOMAD uses a mechanism we term 'ACT-preference' (Granger 1977), which exploits both pragmatic knowledge of what enemies tend to do with weapons, and word-order knowledge that we have derived of how particular triads of prepositions, noun-categories, and verb-categories tend to combine (for example, 'BLAGHED <weapon> AT <ship>' will give rise to a different inference than 'BLAGHED <weapon> TO <ship>', or 'BLAGHED <weapon> FOR <ship>', etc.). This process, detailed in Granger (1977), arrives at an inference that the action is probably a 'PROPEL' (see Schank and Abelson 1977). Again, this is only an educated guess by the system, and may have to be corrected later on the basis of further information (see Granger 1980, 1981b).

Finally, NOMAD produces an interpretation of the input, which the user may or may not confirm. In the event that the user does not confirm NOMAD's initial interpretation, a number of alternative interpretations are produced (see Granger 1981a, 1982c) until one is confirmed, or the process fails. In this and the following examples, NOMAD's 'preferred' interpretation is confirmed by the user.

NOMAD OUTPUT:
An enemy ship fired bombs at our ship.

2. INPUT:
MIDWAY SIGHTED ENEMY. FIRED.

Problem: Missing subject and objects. 'Fired' builds a PROPEL, and expects a subject and objects to play the conceptual roles of ACTOR (who did the PROPELing), OBJECT (what got PROPELed) and RECIPIENT (who got PROPELed at). However, no surface subjects or objects are presented here.

Solution: Use expectations to fill in conceptual cases. NOMAD uses situational (script-based) expectations from the known typical sequence of events in an "ATTACK" – which consists of a movement (PTRANS), a sighting (ATTEND) and

firing (PROPEL) (as in other script-based understanders; see Cullingford 1978). Those expectations say (among other things) that the actor and recipient of the PROPEL will be the same as the actor and direction of the ATTEND, and that the OBJECT that got PROPELed will be some kind of projectile, which is not further specified here.

NOMAD OUTPUT:

We sighted an enemy ship. We fired at the ship.

3. INPUT:

LOCKED ON OPENED FIRE.

Problem: Missing sentence boundaries. NOMAD has no expectations for a new verb ("opened") to appear immediately after the completed clause "locked on". It tries but fails to connect "opened" to the phrase "locked on".

Solution: Assume the syntactic expectations failed because a clause boundary was not adequately marked in the message; assume such a boundary is there. NOMAD assumes that there may have been an intended sentence separation or clause break before "opened", since no expectations can account for the word in this sentence position. Hence, NOMAD saves "locked on" as one clause, and continues to process the rest of the text as a new sentence.

NOMAD OUTPUT:

We aimed at an unknown object. We fired at the object.

4. INPUT:

RETURNED BOMBS TO ENEMY SHIP.

Problem: Multiple word senses of 'returned', resulting in ambiguous interpretation of action. NOMAD cannot tell whether the action here is "returning" fire to the enemy, that is, firing back at them (after they presumably had fired at us), or peaceably delivery bombs, with no firing implied.

Solution: Use expectations of probable goals of actors. NOMAD first interprets the sentence as "peaceably delivering" some bombs to the ship. However, NOMAD contains the knowledge that enemies do not transfer control of weapons, information, personnel, etc., to each other. Hence it attempts to find an alternative interpretation of the sentence, in this case finding the "returned fire" interpretation, which does not violate any of NOMAD's knowledge about goals. It then infers, as in the above example, that the enemy ship must have previously fired on us.

NOMAD OUTPUT:

An unknown enemy ship fired on us. Then we fired bombs at them.

5. INPUT:

OPEN FIRED.

Problem: Lack of tense agreement between 'open' and 'fired'.

Solution: Use morphological analyzer to correct tense of word. NOMAD identifies the phrase 'open fire', and assumes that past tense was intended (by default); and so constructs a phrase that correctly incorporates the tense into the phrase, to make it 'opened fire'. NOMAD then adds the inferred missing actor. (Note that were this not a known phrase to NOMAD then the tense agreement would not have been corrected at the surface level, but rather the semantic content of the two words would have contributed to a meaning representation, which would hae been used to generate a 'corrected' version of the input.

NOMAD OUTPUT:

We fired bombs at an unspecified target.

4.2. Recognizing and correcting interpretation errors

The four interpretation error-types given above were:

1. causal violations,
2. goal violations,
3. user confirmation failure, and
4. out-of-sequence event or object reference.

The process of detecting or correcting these error types is different in principle from the five surface types, for the simple reason that, as opposed to surface errors, which can only be attributed to the message-sender himself, there are many possible different sources of interpretation errors. In particular, some surface errors can give rise to apparent interpretation errors. To see this, recall the 'returned bombs to kashin' example above. In this case, NOMAD's default selection of a word sense for an ambiguous word ('returned') can give rise to an apparent goal violation error (delivery weapons to an enemy, as opposed to firing at an enemy). Hence, the task of blame assignment here is problematic: an early surface-processing decision of NOMAD's can give rise to an apparent later interpretation problem.

Similarly, a 'user confirmation' error (that is, the user will not confirm any of the interpretations offered by NOMAD) might be due to any of a number of things: the user mistyped the original message, NOMAD made an erroneous surface-text decision, or NOMAD failed to detect a surface or interpretation problem in the text. And, a 'causal violation' error (that is, 'ship sighted overhead': ships can't fly, so the error is apparently a user error) can be due either to user errors or to NOMAD's own interpretation errors. Finally, an object or event apparently referenced out of sequence can be due to either user error or an erroneous inference by NOMAD.

5. Summary and Conclusions

5.1. NOMAD's limitations and shortcomings

NOMAD has proved to be a capable analyzer of ill-formed text. Some of the standard problems of script- and plan-based understanders have been satisfactorily addressed in NOMAD, most notably, the handling of unknown words (via the FOUL-UP mechanism); and the script-selection problem, that is, knowing which scripts to apply monitoring when they go wrong (via the mechanisms of supplanting incorrect inferences (Granger 1980), and producing a set of alternate interpretations of a text (Granger 1981a, 198282a, 1982c).

The most important drawback of NOMAD is its lack of extensibility. Since the system's knowledge is mainly embedded in word-level routines, adding a new word to the system requires writing a new routine, possibly duplicating information elsewhere in NOMAD, and possibly introducing new errors into otherwise-working NOMAD code. Any new word routine should ideally take into account interactions with all the word-level routines already present in the system; some of those routines may have to be modified in light of the new entry.

In practice, we do not check every routine when a new word is added. Rather, we test the system and make corrections only when a bad interaction is found. Thus, the system is not guaranteed to be self-consistent. Since NOMAD has more than a thousand-word vocabulary, it is impractical to check the entire system when a new word is added.

Encoding grammatical knowledge at the word level is also cumbersome. For example, the routine for nearly every verb makes its own checks for active or passive usage. A more centralized grammatical mechanism would eliminate this kind of redundancy. In principle, the knowledge currently encoded in the word-level routines could be made declarative (that is, stored as data), so as to be more centralized and usable by other parts of the system.

5.2. VOX: A VOcabulary eXtension system

To make the NOMAD system more extensible, we are currently building a new system that uses not word-level but *phrasal* analysis. We call this new system VOX (for vocabulary extension system). Our goal is to make this system extensible by interaction with a user, rather than by adding to the data base programmatically.

Our ideas about phrasal analysis originate from the work on the PHRAN system (Wilensky and Arens 1982). Phrasal analysis consists of matching the input to one or more phrase-level patterns stored in a knowledge data base. When the input has been matched, it is said to be understood. Semantic actions can be associated with each phrase, so that whenever a phrase is matched to part of the input a corresponding meaning representation for the phrase may be constructed.

To extend the knowledge base of the system, we simply add new patterns to the data base. Ideally, patterns are independent entities whose interaction introduces no side effects, so that new phrases can be easily added to or removed from the data base. A working prototype of VOX is already up and running (see Granger, Meyers, Yoshii, and Taylor 1983 and Meyers 1983), incorporating syntactic and grammatical analyses, semantic analyses and blame assignment, morphological analysis, and error detection and categorization. VOX's phrase knowledge base already consists of hundreds of phrases, and is being extensively tested. Furthermore, VOX's data base can be interactively 'edited' by a trained 'tutor' to add new information, including new vocabulary, new syntactic categories and constructions, and new meanings. Hence, we hope that VOX may be a first step towards a 'trainable' language-processing system. Granger, Meyers, Yoshii, and Taylor (1983) and Meyers (1983) present extensive descriptions of the state of VOX and the theories underlying it.

5.3. Summary: Surface text and its interpretations

The ability to understand text is dependent on the ability to understand what is being described in the text. Hence, a reader of English must have applicable knowledge of both the situations that may be described in texts (for example, actions, states, sequences of events, goals, methods of achieving goals, etc.), and the surface structures that appear in the language, that is, the relations between the surface order of words and phrases, and their corresponding meaning structures. The process of text understanding is the combined application of these knowledge sources as a reader proceeds through a text. This fact becomes clearest when we investigate the understanding of ill-formed texts, texts that present particular problems to a reader. The line between correct and incorrect English is often unclear, so a system that cannot handle erroneous input is of limited use.

Human understanding is inherently tolerant; people are naturally able to ignore and deal with many types of errors, omissions, poor constructions, etc., and get straight to the meaning of the text. Our theories have tried to take this ability into account by including knowledge and mechanisms of error noticing and correcting as implicit parts of our process models of language understanding. NOMAD and VOX are primarily engineering applications incorporating a series of theoretical results in language understanding, including script-based and goal-based understanding, and integrated error-monitoring and supplanting during understanding. The NOMAD and VOX systems are the lat-

est in a line of 'tolerant' language understanders, beginning with FOUL-UP, all based on the use of knowledge of syntax, semantics, and pragmatics at all stages of the understanding process to cope with errors.

Acknowledgments

Rika Yoshii, Chris Staros, Greg Taylor, and Amnon Meyers all contributed to the construction of the NOMAD system; Amnon Meyers has been responsible for the construction of VOX.

References

Birnbaum, L. and Selfridge, M. 1980 Conceptual Analysis of Natural Language. In Schank, R. and Riesbeck, C., Eds., *Inside Computer Understanding.* Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Cullingford, R. 1977 Controlling Inferences in Story Understanding. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI).* Cambridge, Massachusetts.

DeJong, G. 1979 Skimming Stories in Real Time: An Experiment in Integrated Understanding. Ph.D. Thesis. Computer Science Department, Yale University, New Haven, Connecticut.

Goldman, N. 1973 The Generation of English Sentences from a Deep Conceptual Base. Ph.D. Thesis. Stanford University, Stanford, California.

Granger, R.H. 1977 FOUL-UP: A Program that Figures Out Meanings of Words from Context. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI).* Cambridge, Massachusetts.

Granger, R.H. 1980 When Expectation Fails: Toward a Self-Correcting Inference System. *Proceedings of the First National Conference on Artificial Intelligence.* Stanford University, Stanford, California.

Granger, R.H. 1981a Directing and Re-directing Inference Pursuit: Extra-textual Influences on Text Interpretation. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence (IJCAI).* Vancouver, British Columbia.

Granger, R.H. 1981b Shaping Explanations: Effects of Questioning on Text Interpretation. *Proceedings of the Third Annual Conference of the Cognitive Science Society.* Berkeley, California: 193-196.

Granger, R.H. 1982a Judgmental Inference: Inferential Decision-Making during Understanding. Technical Report #182. Computer Science Department, University of California, Irvine, California.

Granger, R.H. 1982b Scruffy Text Understanding: Design and Implementation of 'Tolerant' Understanders. *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics.* Toronto, Ontario, Canada: 157-160.

Granger, R.H. 1982c Inference Decisions in Text Understanding. *Proceedings of the Fourth Annual Conference of the Cognitive Science Society.* Ann Arbor, Michigan.

Granger, R.H.; Meyers, A.; Yoshii, R.; and Taylor, G. 1983 An Extensible Natural Language Understanding System. *Proceedings of the Artificial Intelligence Conference.* Oakland University, Rochester, Michigan.

Hayes, P.J. and Mouradian, G.V. 1981 Flexible Parsing. *American Journal of Computation Linguistics* 7(4): 232-242.

Kwasny, S.C. and Sondheimer, N.K. 1981 Relaxation Techniques for Parsing Grammatically Ill-Formed Input in Natural Language Understanding Systems. *American Journal of Computation Linguistics* 7(2): 99-108.

Lebowitz, M. 1981 Generalization and Memory in an Integrated Understanding System. Computer Science Research Report 186. Yale University, New Haven, Connecticut.

Meyers, A. 1983 Conceptual Grammar. Computer Science Technical Report #215. University of California, Irvine, California.

McGuire, R. 1980 Political Primaries and Words of Pain. Unpublished manuscript. Department of Computer Science, Yale University, New Haven, Connecticut.

Riesbeck, C. and Schank, R. 1976 Comprehension by Computer: Expectation-based Analysis of Sentences in Context. Computer Science Research Report 78. Yale University, New Haven, Connecticut.

Schank, R.C. and Abelson, R. 1977 *Scripts, Plans, Goals, and Understanding.* Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Small, S. 1980 Word Expert Parsing: A Theory of Distributed Word-Based Natural Language Understanding. Technical Report TR-954. University of Maryland, College Park, Maryland.

Wilensky, R. 1978 Understanding Goal-Based Stories. Computer Science Technical Report 140. Yale University, New Haven, Connecticut.

Wilensky, R. and Arens, Y. 1982 PHRAN: A Phrasal Analyzer. EECS Technical Report. University of California, Berkeley, California.

APPENDIX: Some Statistics on NOMAD's Operation

1. *Timing:* NOMAD uses about 3 cpu seconds per word when analyzing Navy messages.
2. *Vocabulary size and structure:* NOMAD is based on CA (Birnbaum and Selfridge 1979), and incorporates 'word-expert' routines (Small 1980). Each word-expert routine can process a whole class of words, not just an individual word. There are 152 word-expert routines; there are 440 words, inflected forms, and phrases in NOMAD's dictionary. (There are 330 words and phrases, not counting inflections.)
3. *Knowledge:* There are 16 situation frames, corresponding roughly to: battle, communication, location-change, sight, attack, report, command, communicate, emanate, detect, project, aim, ptrans, patrol, state-change, causal-result.
4. *Benchmarks:* NOMAD has successfully processed about 4000 Navy messages of lengths varying from 1 line to 17 lines of text each. No statistics have been compiled on NOMAD's overall success versus failure rate on all Navy texts.