A Knowledge-Rich Approach to Understanding Text about Aircraft Systems

Peter Clark, Lisbeth Duncan, Heather Holmback, Tom Jenkins, John Thompson
Boeing Engineering and Information Technology
Boeing, Seattle, WA 98124
{peter.e.clark,lisbeth.e.duncan,heather.holmback,thomas.l.jenkins2,john.a.thompson}@boeing.com

ABSTRACT

As part of a longer-term goal to construct an aerospace knowledge-base (KB), we are developing techniques for interpreting text about aircraft systems and then adding those interpretations to the KB. A major challenge in this task is that much of what is written builds on unstated, shared, general knowledge about aircraft, and such prior knowledge is needed to fully understand the text. To address this challenge, we are using a more general KB about aircraft to create strong, prior expectations about what might be stated in that text, then treating the language understanding task as one of incrementally extending and refining that prior knowledge. The KB constrains the possible interpretations of the text, allowing it to be placed in the appropriate context and helping identify when statements can be taken literally or need to be coerced or modified to be understood correctly. In this paper we present this approach and discuss its underlying assumptions and range of applicability. The significance of this work is twofold: It illustrates the critical role background knowledge plays in fully understanding language, and it provides a simple model for how that understanding can take place, based on the iterative refinement of a representation using information extracted from text.

1. OVERVIEW

Our long-term goal is the construction of a large-scale aerospace knowledge-base (KB), for use in tasks such as concept-based information retrieval (described in [6]) and automated question-answering about aircraft. To help construct this KB, we are developing language processing techniques to interpret text about aircraft [14, 12, 5], e.g., from existing training manuals authored in a restricted version of English or directly from an aerospace engineer. The resulting interpretation is then added to the KB.

A major challenge in this task is that much of what is written builds on unstated, shared, general knowledge about aircraft, and such prior knowledge is needed to fully understand the text. In particular, a literal translation of the text is often either highly incomplete, or worse incorrect, if no further processing is done. For example, consider:

(1) The hydraulic system supplies power to the rudder.

.

Correctly interpreting this sentence is surprisingly difficult: here the author's intent was not to assert that some supplying event is currently ongoing, but to refer to a *typical behavior* of the hydraulic system in action. Moreover, in writing this sentence, the author is appealing to some unstated, shared, general knowledge: Both the author and intended reader already know that a rudder is a powered device, that a powered system typically consists of a power source and a powered device, that the source transmits power via some conduit, etc. The author intended this sentence to build upon this knowledge, so that the reader realizes that what (1) is really describing is a *powered system* within the aircraft (as opposed to just an isolated supplying event), and hence that the many implications of this realization can be inferred. For automated language processing, this ability to interpret text as specializing, refining, and building upon background knowledge is essential for both acquiring the full meaning of text, and for determining which statements can be taken literally or need to be coerced, modified, or set in context.

To address this challenge, we are constructing and using an extensive, but fairly general, knowledge-base about aircraft, called AeroNet [5], to provide the knowledge and expectations required to properly interpret the text. The KB contains general knowledge about aircraft, but does not contain the many detailed facts about specific airplanes or their internal systems. It is this latter knowledge that we wish the system to acquire from text.

In particular, the knowledge base and its underlying inference engine allow the system to construct and incrementally refine a representation of an aircraft or an aircraft subsystem. For example, a (representation of a) specific airplane having n engines may be refined to one having two engines, to one having two jet engines, to one having two jet engines connected to the wings, etc. The mechanism for doing this is structure unification [4] (similar to order-sorted feature unification [2]), with a corresponding semantics in first-order logic. One can think of an initial, general, representation of an airplane as denoting a space of possible airplanes, and the language processing task is to incrementally refine this representation, by fitting (coercing and unifying) the processed text into that representation.

This use of domain-specific, background knowledge to form strong expectations about what might be said in text has similarities with the use of scripts and templates (e.g., [7, 8]) in language interpretation. In a similar way, scripts encode shared, unstated background knowledge, providing the context in which knowledge from text should be placed, and enabling question-answering to go beyond the facts directly stated. What is new here is the treatment of this process as incremental refinement of a representation, rather than "filling in the blanks." Rather than artificially dividing the world into 'known' and 'unknown' facts (as in a template) and then hunting for the unknown, we instead just have levels of approximation, and any part of the representation can be be refined through a process of logical unification. Similar ideas have been explored in the context of constraint logic programming [1]. This approach can also be seen as a particular implementation of the general "interpretation as abduction" framework [13], in which the abductive inferences are unifications (equality assertions) of background knowledge with knowledge from text.

2. APPROACH

2.1 Scope of the Problem

The scope of the problem we are addressing is constrained in several in important ways. First, we are processing text authored in an informally restricted sublanguage of English, rather than full natural language, mainly consisting of full, declarative sentences and without excessive syntactic complexity (authored in this way to enable ease of understanding, in particular for readers for whom English was not their native language). Second, we are working with a constrained domain (aircraft systems), which significantly reduces the scope of the target vocabulary, problems such as word ambiguity, and the amount of knowledge required up front in the AeroNet KB. Our initial work has been further constrained to the subdomain of airplane hydraulic systems, and as data we have been using text taken from three pages of a hydraulics training manual. Third, the nature of this particular application domain itself simplifies the problem: as it essentially concerns mechanical artifacts, we are primarily dealing with text describing the physical structure and behavior of mechanical artifacts, as opposed to (for example) text describing human affairs, or discussing points of view. As a result, the text we are processing uses a constrained vocabulary and is simple, regular, and (from a literary point of view) "boring" in style, and poses fewer representational challenges than might otherwise be the case. Finally we note that the background AeroNet knowledge base itself was already partially constructed from an earlier project, and is fairly large (500 concepts, 1500 axioms). We have further extended it for this work.

2.2 The Knowledge Base

The background KB, AeroNet, is encoded in the representation language KM, a frame-based representation language similar to KRL [3] and with first-order logic semantics [4]. AeroNet contains an ontology (class hierarchy) of aerospace and more general concepts, and axioms about those concepts. The majority of its axioms represent, at a general level, the possible configurations of physical objects, events, and their relationships for aircraft, aircraft components, and behaviors associated with those components. In terms of logic, they assert, for a given concept, the existence of additional objects and relationships associated with that concept (e.g., for all airplanes there will exist a fuselage which is part of that airplane). These axioms are important because they are the building blocks from which representations of a possible, specific aircraft are constructed, which in turn form the the possible targets for interpreting natural language statements (i.e., a statement is understood by identifying which of these possible aircraft configurations it is referring to).

A critical feature of the KB, for this application, is the way these axioms are physically encoded. As well as supporting normal logic axioms, KM also supports the use of *prototypes* as an alternative syntax for expressing the common "forall...exists..." axiom form. In KM, a prototype is a graph data structure describing an example of a concept, and is formed by Skolemizing the existentially quantified variables in the original axiom (thus reducing it to a set of ground assertions between Skolem constants). KM's inference engine handles prototypes in a special way, by asserting that the those assertions holds for *all* instances of that concept (thus applying the same semantics as the original axiom). An illustration of these encodings is shown in Figure 1. More details are of KM's prototype mechanism are available in [4]. (Prototypes have been used in other AI languages also, e.g., in KRL [3]).

KM Axiom (standard form)

```
(every Airplane has
  (parts ((a Fuselage)
         (a Tail with
              (connected−to
                  ((the Fuselage parts of Self))))
```

KM Axiom (prototype form)

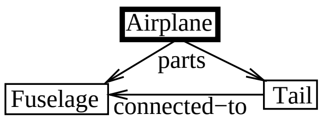


Figure 1: Axioms are converted to a prototype form, encoded using a graph data structure, to enable matching with NLP-generated graph structures. This figure shows the axiom "All airplanes have a fuselage and a tail connected to that fuselage." in both forms.

The significance of prototypes is that they form the critical bridge from logical axioms in the KB to the graph-like structures output by our NL processor. By syntactically reformulating sentence-like axioms as prototype graphs, the task of aligning text with axiom-encoded knowledge becomes one of aligning, combining, and reasoning over graphical data structures. Other work in NLP has similarly found that encoding axioms using graphical data structures is helpful for the same reasons, e.g., [16].

2.3 Overview of the Process

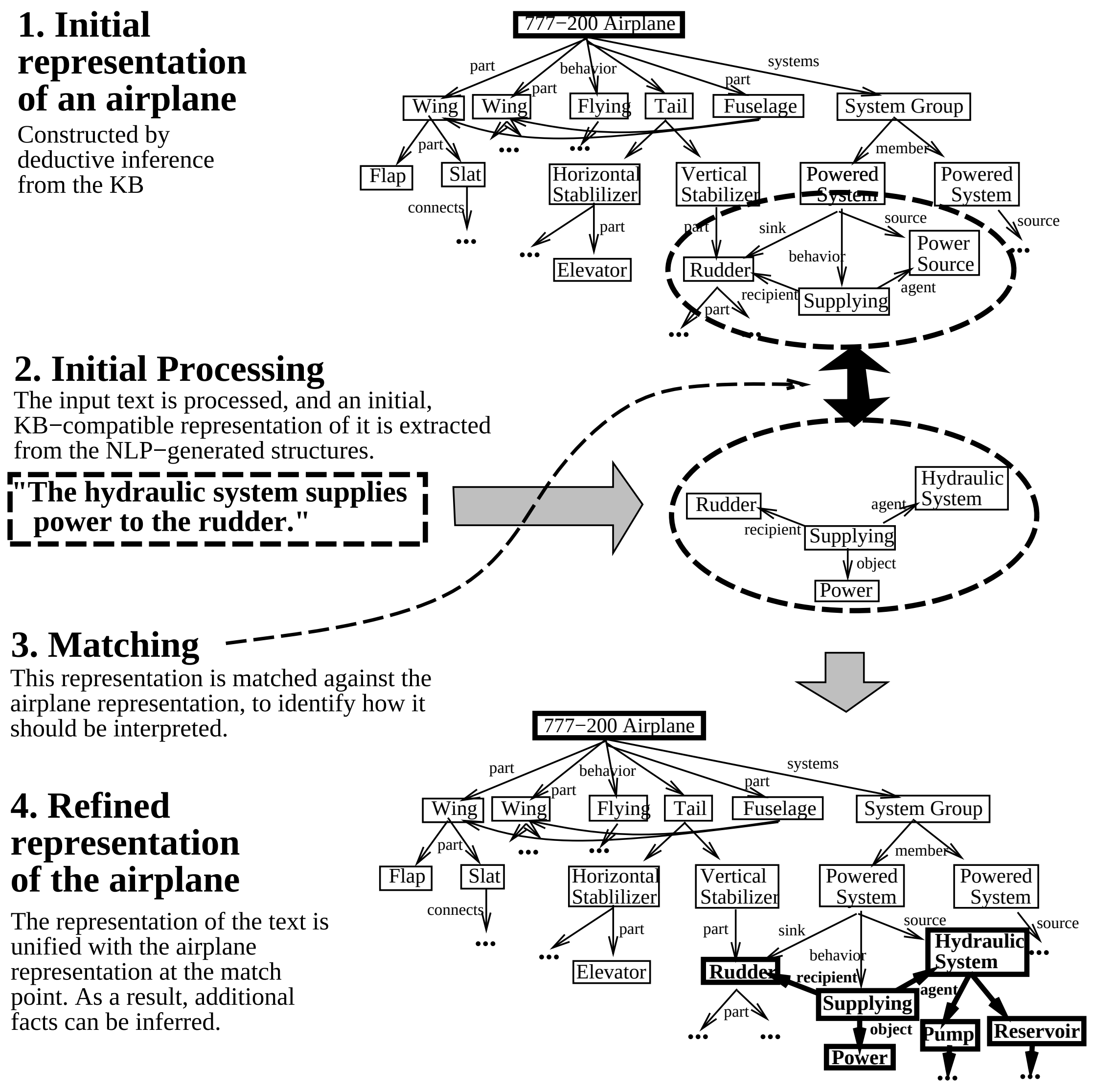
Before processing a unit of text, the user first manually specifies the object ("topic") the text describes (either a specific airplane or airplane system). The system then creates an initial, general representation of that airplane/airplane system using the KB (Step 1 of Figure 2).
Processing the text itself then occurs in two stages as follows:

- First, a unit of text is processed to construct an initial, literal, or sometimes underspecified, representation of the text.
- Second, the system searches for a place where this structure can "fit in" with the current representation of the airplane/airplane subsystem, and when one is found, the structure is unified at that point, hence specializing that representation.

This second stage could be described as "model-based interpretation" of text, as it treats language understanding primarily as a process of specializing, instantiating, and combining pre-existing knowledge structures, guided by the content of text. The pre-existing structures provide the reference point for interpreting the input language, allowing determination of what statements can be taken literally, and what statements need to be to be coerced, modified, or set in context.

2.4 Stage 1: Initial Interpretation and Normalization

Consider that the user has stated that the topic is the 777-200 airplane, and entered the aircraft-related sentence discussed earlier:


1. Initial representation of an airplane
Constructed by deductive inference from the KB

2. Initial Processing
The input text is processed, and an initial, KB−compatible representation of it is extracted from the NLP−generated structures.

"The hydraulic system supplies power to the rudder."

3. Matching
This representation is matched against the airplane representation, to identify how it should be interpreted.

4. Refined representation of the airplane
The representation of the text is unified with the airplane representation at the match point. As a result, additional facts can be inferred.

Figure 2: Text is understood by matching a KB-compatible representation of it, extracted from initial language processing, against a representation containing general expectations (derived from background knowledge). As a result, that representation is refined to absorb the new knowledge. (The graphs shown here are much simplified).

(1) The hydraulic system supplies power to the rudder.

(This sentence contains new knowledge as in general rudders can be powered in other ways also, e.g., using mechanical cables, and not all airplanes have hydraulic systems. Although these general possibilities are pre-encoded in the KB, the KB does not know which of these are realized in the 777-200 airplane. It is this knowledge which the sentence conveys and which we wish the system to acquire).

The goal of the first stage in processing is to generate an initial interpretation of this text, expressed in terms of AeroNet's ontology (Figure 3). To do this, the input text is processed using Boeing's natural language understanding system. This system uses a bottom-up chart parser interleaved with semantic interpretation, and produces a full syntactic analysis paired with a semantic analysis showing the relations between the elements of the sentence. Word senses relevant to the domain are also distinguished in this stage [15, 10, 9]. Following this, a set of simple manipulation rules are used to rewrite the relevant information from the resulting structure into a form consistent with AeroNet's pre-built ontology (Step 2 of Figure 2). These rules build a representation of events, entities, locations, properties, etc., in the input, and synthesizes information across the sentences in cases of a multi-sentence unit of text. Contrary to the other applications where we include a large degree of detailed domain inferencing in the sentence-level semantic and discourse-level interpretation phase, in this work the natural language understanding system is designed to produce a more superficial, less specific interpretation of the input text.

One issue with a "pipelined" system like this is how much disambiguation and filling in of information should be done at this stage. In this case, the system attempts to choose an overall parse/syntactic configuration, but may not be able to fill in all the semantic relationships between the constituents. For example, the system will not commit to how elements of a 3-item noun compound are grouped together, or what the semantic relations among the elements are.

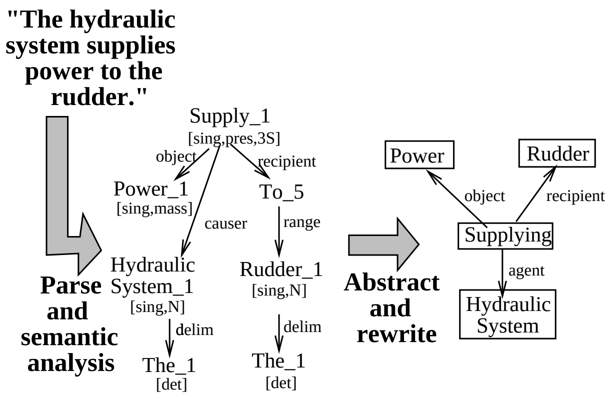


Figure 3: During the first stage of processing, the text is parsed and an NLP-generated structure (logical form) is generated. Following this, simple rewrite/renaming rules convert this structure into one expressed in terms of AeroNet's ontology.

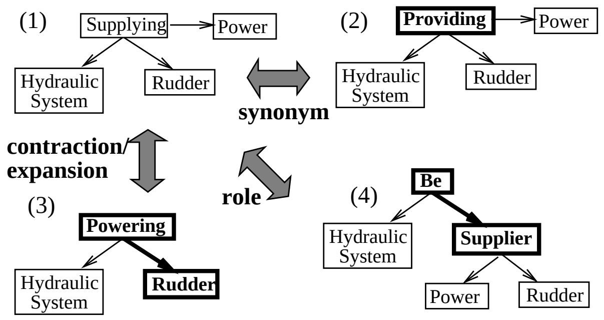


Figure 4: If the initial NLP-generated structure fails to match the KB, the system tries various transformations and re-attempts matching, in order to accomodate alternative ways of expressing the same knowledge. Structures (2)-(4) show alternative structures expressing the same knowledge as in (1) (see the body of the paper for their English equivalents).

This remains underspecified at this point, to be resolved later using the KB in stage 2. Although there is of course the risk of mistakes in this stage, the risk is partially reduced due to the constrained application domain and sublanguage targetted. Furthermore, the NL system does include a large amount of information on verbs that are likely to occur in the application domain, based on many years of development to handle maintenance manuals and other aerospace documents [14, 12].

2.5 Stage 2: Integration with Existing Knowledge

The goal of the second stage is to correctly integrate this statement with the current representation of the topic airplane. To do this, the system searches the airplane representation, including possible extensions of that representation, for (in this case) candidate "supplying" events that the user may be referring to. This is illustrated in Step 3 of Figure 2. A matching event is one that is equal or more general than (i.e., subsumes) the event described in the text. If multiple matches are found, then a (currently naive) scoring system determines the best match. In addition, it will also search (and again possibly extend) the airplane representation for the objects playing roles in that event.

If a match is found, then the text representation is unified with the airplane representation at the match point, thus adding details to the airplane representation which were specified in the text. In graphical terms, unification involves maximally merging the two graphs, where two arcs can merge if they denote the same relationship, and two nodes can merge if one subsumes (is more general than) the other. In logical terms, unification involves recursively asserting equality between objects in the text representation and Skolem instances in the current airplane representation, subject to the same constraints. This is illustrated in Step 4 of Figure 2.

For this matching to succeed in this straightforward way, the structure representing the user's statement, produced in stage 1, must correspond exactly with the representational structures used in the KB. However, in practice this condition may not hold for two reasons:

Linguistic Variation: Any piece of knowledge can be expressed in language in a variety of different ways, resulting in a variety of different structures being output by the stage 1 processing, only some of which may neatly correspond to the structures in the KB. Note that although the stage 1 processing removes some of these variations by converting the input text into a more normalized form (e.g., by converting active and passive sentences into the same output structure), it is not possible to remove all variations at this stage.

Expression of Derived Facts: The representations explicit in the KB only describe a subset of the things that a user might state or refine; on top of this, there are also a large number of "derived facts" or "viewpoints" on that knowledge which the user could also potentially refine, and which the matcher needs to have access to if it is to also recognize those user inputs.

To deal with these phenomena, if the original matching fails then the system will try to *coerce* (modify) the text representation to match, by applying simple transformation rules to both the text representation and the target KB structures during matching. This enables the system to also understand text expressing the same knowledge but in alternative ways, and to understand statements concerning the wider body of facts implied by, but not explicitly stated in, the KB structures.

For linguistic variation, the matcher currently tries three types of transformations to modify the text representation, namely the replacement of concepts by near-synonym concepts, expanding verb-object contractions (and vice versa), and removing/adding explicit mention of roles. (There are also other dimensions of linguistic variation which we have not yet accounted for, which we summarize later.). As illustration, these allow the (structures corresponding to) the below sentences to be transformed into each other. In particular, the transformations (1) $\leftrightarrow$ (2), (1) $\leftrightarrow$ (3), and (1) $\leftrightarrow$(4) illustrate the near-synonym, verb-object contractions, and role transformations respectively, illustrated in Figure 4:

(1) The hydraulic system supplies power to the rudder.
(2) The hydraulic system {provides/sends/transmits/gives} power to the rudder.
(3) The hydraulic system powers the rudder.
(4) The hydraulic system is the {supplier/provider} of power for the rudder.

By applying these transformations, alternative structures denoting the user's input are derived and matched with the KB if the original structure fails to match. Note that none of the (structures corresponding to) these sentences is considered to be the "canonical representation" of the input text; rather, the transformations allow these alternative structures to also be tried when searching for a match in the KB. This is important, as the structure of the corresponding knowledge in the KB itself depends on domain-specific design decisions made when building the KB, may vary depending on the particular concepts involved, and may change if the KB itself is later redesigned to change those design decisions. It also allows the stage 1 NLP system to defer application-dependent decisions about (for example) which words are synonyms, and thus remain general-purpose (as opposed to it "hard wiring" those decisions into it, which would make it highly domain-specific).

Concerning derived facts, the matcher needs to account for user statements which may refer to an *implication* of knowledge in the KB, rather than just to knowledge explicitly represented in (i.e., as subgraphs of) the KB prototypes. To do this, it needs to match against an extended version of the KB, extended to contain those implications. This extended KB is not generated in its entirity (it would be impractical to do so); rather it is a "virtual KB" giving the appearance of being there to the matcher, while in practice parts are generated on demand. To generate these extensions, we are currently using just three general rules, although clearly more are needed:

1. "Causal transitivity": If X is the agent in event E, then add a representation of X as the agent in all the causally downstream events of E.
2. "Object as instrument": If X does E to Y, causing Y to do E' to Z, then add a representation of Y as the instrument in X doing E' to Z.
3. "System for system part": If X does E, and X is a part of system Y, then add a representation of Y as doing E.

As a (simplified) example of the first two rules being applied, consider Figure 5. If the KB explicitly contains a representation of the causal chain (for a cable-driven rudder control system's behavior):

(5) The pilot presses a foot pedal.
(6) The foot pedal pulls a cable.
(7) The cable moves the rudder.

Then the virtual KB will also contain representations of:

(8) The pilot moves the rudder using the foot pedal.
(9) The pilot moves the rudder using a cable.
(10)The pilot pulls a cable using the foot pedal.
(11) The foot pedal moves the rudder using a cable.

As a result, the user's statement is matched against all these target structures, thus accommodating statements which refine implicit as well as explicit information in the KB. Some of these derived representations are abstractions of knowledge in the KB, while some are alternative viewpoints on the knowledge in the KB. Also, in a few cases, the derived representation can violate KB constraints, i.e., do not make sense when taken literally (e.g., "The pilot consumes fuel," from "The pilot operates the engines." and "the engines consume fuel."). Rather than discard these, they can be viewed as KB-generated metonymous representations (i.e., where an object stands in place of a closely related object), and used to recognize and correctly interpret similar uses of metonymy in the input text. In this case, the inference rules are playing the role of metonymic transformation rules (similar to those used by Fass [11]).

Returning to the running example (1), as AeroNet already knows that airplanes (in general) contain rudders and some powered system must drive them, the search will find the supplying event that is the behavior of that powered system. As this matches (subsumes) the input text, the text representation is unified with this supplying event, resulting in the rudder's power source being refined to be

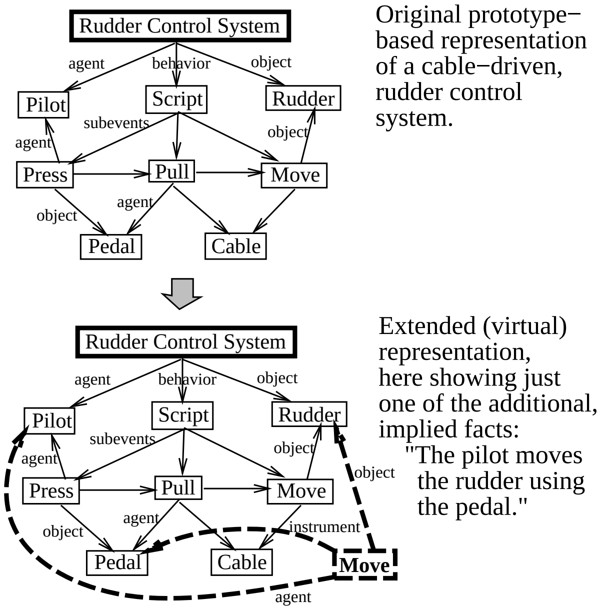


Figure 5: To handle statements referring to facts not explicit in the original representation, the matcher matches against a (virtual) extended version of the representation, extended to also include implied facts.

a hydraulic system. From here, now that the system knows a hydraulic system is in the airplane, additional facts can be inferred (i.e., the graph grown further) using knowledge about hydraulic systems in the KB. This is illustrated in Step 4 of Figure 2.

The significance of this stage 2 processing is that, rather than simply (and erroneously) assert the existence of this supplying event, the system has determined the appropriate context in which that statement belongs, and added it at that point. As a result, a fuller meaning of the text (namely that it is implicitly describing part of the characteristic behavior of rudder's power system) has been acquired by the system, even though that extra information has not been explicitly stated in the text.

3. STATUS AND DISCUSSION

While the stage 1 NL processor is mature and operates on a broad range input language, and the AeroNet KB is also large (500 concepts, 1500 axioms), our work on their integration is still preliminary and we have only achieved full start-to-end throughput on a small number of sentences. Part of the reason for this is simply the implementation not keeping up with the theory, but part of it is due to more fundamental challenges of text understanding which our approach cannot currently handle. In this section, we attempt to distinguish these two aspects by characterizing the scope of the presented approach, and cases where it breaks down.

Our approach relies on several mechanisms and assumptions. First, it relies on the KB creating strong expectations about what knowledge might be expressed, and on the text's author conforming to those expectations. In addition, it relies on the language processing and matching processes to tolerate variations in how that knowledge might be expressed. For the matching process to work, the structure of the processed language (the output of stage 1) must be correct and reasonably close to the structures in the KB, and the sentences need sufficient detail to avoid excessive ambiguity when matching with the KB. To meet these requirements, the input sentences need to be fairly context-independent, accurate, and have a relatively simple structure, for example containing short, declarative statements about properties of objects, relationships between objects, and simple descriptions of events. (The majority of sentences in the training manual text we have been using meet these requirements). The primary challenge has not been in the stage 1 language processing itself, but in bridging the gap between its result and the structures in the KB. An important contribution of this work has been allowing a wider gap to be tolerated through the application of transformation and inference rules in stage 2 (e.g., allowing some types of metonymy to be identified and corrected), but still the gap must not be too great for the described approach to be effective.

There are several phenomena (linguistic and otherwise) which can occur which cause these assumptions to be violated, including:

1. Inaccurate/simplified knowledge. If the user is inaccurate or deliberately simplifies in the knowledge, resulting structures may fail to match those in the KB. For example,

   (12) The pump supplies a source of power to the rudder.

   is, strictly speaking, incorrect (the pump supplies power, not a source of power), and hence will fail to match knowledge in the KB, even using transformations. While here the author may simply have been "sloppy" accidentally, in other cases he/she may deliberately make technically inaccurate statements so as to simplify for pedagogical purposes.
2. Fluctuating ("fuzzy") concept boundaries. Sometimes the precise boundaries of a concept may subtly vary in text. For example, does the "hydraulic system" include its controlled devices (e.g., the rudder) or not? Some text treats the answer as yes, other text as no, in contrast to the KB which has a single, precise answer to this question, and cannot cope if the user deviates from that. Strictly, the user's variations correspond to (subtly) different senses of the phrase "hydraulic system", but it is unrealistic to treat them as a (potentially large) number of different word senses to be disambiguated.
3. Ambiguities which are not locally resolvable. Although the KB significantly restricts ways in which the input text will be understood, there still may be ambiguity (i.e., multiple matches with the KB). For example, in

   (13) The EDP is on the right side of the airplane's engine.

   the reader understands "the airplane's engine" as "the airplane's propulsion engine," although to the matcher it is ambiguous (there are other engines in an airplane besides its propulsion engines). In these cases, the author is additionally relying on context and unstated communication protocol to identify the preferred match, which we have not accounted for.
4. Complex sentence structure. We assume that the organization of constituents of the input sentences will be correctly determined by the stage 1 processing. However this becomes harder to achieve as sentences get longer and more ambiguous. A (rather extreme) example is:

   (14) The RAT blade lock pin behind the turbine locks the turbine blades in a vertical position when the RAT is more than 9 degrees from the extended position.

   As sentences become more complex, it becomes harder for the stage 1 NL system to build a complete semantic representation with all the information packaged so that it can be used easily by the KB.
5. Other linguistic variations. Although the NLP processor will normalize many verb alternations to a common structure, and the transformation rules account for some additional linguistic variations in expressing knowledge, there are still variations which have not been covered, and hence will not be interpreted by the system.

An additional challenge occurs when multiple transformations of the text and KB are required to achieve a match, and this has been the main obstacle to processing multi-sentence input. The challenge here is not so much in generating a single, integrated structure representing the multi-sentence text (which the stage 1 NL processor can do effectively), but in then matching that structure against the KB. While small sentence-based structures can be feasibly transformed to achieve a match, the task becomes highly complex with paragraph-equivalent sized structures. To deal with longer text units, better interleaving of the stage 1 and stage 2 processing is clearly desirable.

4. SUMMARY AND CONCLUSION

We have presented an approach to understanding text about aircraft systems, based on using a KB to create strong, prior expectations about what might be stated in that text, and on treating the language understanding task as one of incrementally refining an airplane representation build using this prior knowledge. The KB constrains the possible interpretations of the text, allowing it to be placed in the appropriate context and helping identify when statements can be taken literally or need to be coerced or modified to be understood correctly. We have also described the assumptions on which this approach is based and which determine its range of applicability.

As a practical means of extending a knowledge-base, it is worth reiterating several factors that simplify the task and make it more feasible: we are working with training manuals, whose language is deliberately simplified (as opposed to technical maintenance manuals, whose language can be more complex); the stage one NLP system has been developed and used to process manuals (for other purposes) for several years; the domain is constrained; and we are not attempting to extract every item of information from the text – rather, the system will simply ignore text that does not match any expectations in the airplane model being constructed, and hence we are not dependent on fully understanding the entire text in order to acquire at least some knowledge using this method (the system will extract what it can, and simply ignore the rest). As an additional benefit, this method opens the possibility of knowledge entry directly from an engineer using a controlled language interface, which we are also exploring, rather than requiring him/her to learn and work with the underlying knowledge representation language.

One of the most difficult challenges has been determining the relationship between the "stage 1" and "stage 2" processing steps, and how much decision-making (e.g., word sense disambiguation, normalization of the logical forms) should be performed during initial, KB-independent text processing (stage 1), and how much with reference to the KB (stage 2). Too much, and the system may make premature and erroneous commitments up front; too little and there may not be enough structure identified for the knowledge-rich processing to operate. In practice, we have deliberately chosen not to exploit the full power of the stage 1 NL processor, instead delaying some decisions (e.g., normalizing the earlier variants (2), (3) and (4)) to the second processing stage, thus avoiding redundantly hard-wiring the required knowledge in the stage 1 lexicon and normalization rules. Similarly, there is clearly scope for the KB to provide additional guidance during the initial processing. These issues are critical for any knowledge-based NLP approach to address.

Although subject to a number of constraints, the approach suggests how a strong interplay between language processing and background knowledge can be achieved. The significance of this work is twofold: it illustrates the critical role background knowledge plays in fully understanding language, and provides a simple model for how that understanding can take place, based on the iterative refinement of a representation using information extracted from text.

4.1 Acknowledgements

Many thanks to Ken Barker and Gary Coen for valuable feedback on earlier versions of this paper.

5. REFERENCES

[1] H. Ait-Kaci and A. Podelski. Towards a meaning of LIFE. *Logic Programming*, 16:195–234, 1993. (also available as http://www.isg.sfu.ca/ftp/pub/hak/prl/PRL-RR-11.ps.Z).

[2] H. Ait-Kaci, A. Podelski, and S. C. Goldstein. Order-sorted feature theory unification. In D. Miller, editor, *Proc. Int. Symp. on Logic Programming*, pages 506–524. MIT, 1993.

[3] D. Bobrow and T. Winograd. An overview of KRL, a knowledge representation language. In R. Brachman and H. Levesque, editors, *Readings in Knowledge Representation*, pages 264–285. Kaufmann, CA, 1985. (originally in Cognitive Science 1 (1), 1977, 3–46).

[4] P. Clark and B. Porter. KM – the knowledge machine: Users manual. Technical report, AI Lab, Univ Texas at Austin, 1999. (http://www.cs.utexas.edu/users/mfkb/km.html).

[5] P. Clark, J. Thompson, H. Holmback, and L. Duncan. AeroNet: Building the aeronet knowledge base. Technical report, Boeing Mathematics and Computing Technology, Seattle, WA, 2000.

[6] P. Clark, J. Thompson, H. Holmback, and L. Duncan. Exploiting a thesaurus-based semantic net for knowledge-based search. In *IAAI-2000*. AAAI Press, 2000. (http://www.rt.cs.boeing.com/∼clarkp/papers/iaai00.ps.Z).

[7] R. E. Cullingford. Controlling inference in story understanding. In *IJCAI-77*, page 17, 1977.

[8] G. DeJong. Prediction and substantiation: two processes that comprise understanding. In *IJCAI-79*, pages 217–222, 1979.

[9] L. Duncan, W. Brown, C. Esposito, H. Holmback, and P. Xue. Enhancing virtual maintenance environments with speech understanding. Technical Report TECHNET-9903, Boeing Mathematics and Computing Technology, 1999.

[10] L. Duncan, P. Harrison, H. Holmback, T. Jenkins, A. Kao, S. Miller, S. Poteet, and J. Powell. Message processing and data extraction for the real time information management system (RTIMS) project. Tech Report BCSTECH-94-057, Boeing Computer Services, 1994.

[11] D. Fass. met*: A method for discriminating metonymy and metaphor by computer. *Computational Linguistics*, 1(17):49–90, 1991.

[12] J. Hoard, R. Wojcik, and K. Holzhauser. An automated grammar and style checker for writers of simplified english. In P. Holt and N. Williams, editors, *Computers and Writing: State of the Art*, pages 278–296, Oxford, England, 1992. Intellect.

[13] J. Hobbs, M. Stickel, D. Appelt, and P. Martin. Interpretation as abduction. *Artificial Intelligence*, 63(1–2):69–142, 1993.

[14] H. Holmback, L. Duncan, and P. Harrison. A word sense checking application for simplified english. In *Third International Workshop on Controlled Language Applications*, 2000.

[15] T. A. Nicolino. A natural language processing based situation display. In *Proc. 1994 Symposium on Command and Control Research and Decision Aids*, pages 575–580, Naval Postgraduate School, Monterey, CA, 1994.

[16] J. F. Sowa. *Conceptual structures: Information processing in mind and machine.* Addison Wesley, 1984.