Technical Report

UCAM-CL-TR-17
ISSN 1476-2986

Number 17

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Three papers on parsing

B.K. Boguraev, K. Spärck Jones, J.I. Tait

1982

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

This work is supported by the Science and Engineering Research Council and by the British Library Research and Development Department

So what about parsing compound nouns?

Karen Sparck Jones
Computer Laboratory, University of Cambridge
Corn Exchange Street, Cambridge CB2 3QG

The problem

Compound nouns, and more generally, complex nominals, are a problem we have turned away from. But any serious natural language program, whether practically or theoretically motivated, has got to tackle it.

Three processes apply to compound nouns. They have to be bracketed: 'wicker (bread basket)' vs '(wholemeal bread) basket'; they have to be lexically disambiguated: 'bread = loaf' vs 'bread = money'; and they have to be given a meaning characterisation: 'basket for bread' vs 'basket with bread'.

Various strategies, or rather non-strategies, have been adopted to deal with compound nouns. One, for example, is to put compound nouns in the lexicon. A second is to assume that restrictions on the possible interpretations of the constituents, for a specific universe of discourse, will force an appropriate interpretation on the whole. A third is not to attempt a full characterisation of the whole. And a fourth is to disregard elements which cannot be processed. These are all unsatisfactory, for obvious reasons. We cannot put all the compounds, even for a limited domain, in the lexicon. We cannot rely on sense restrictions to give us an acceptable interpretation. We cannot assume that an explicit characterisation is unnecessary. And we cannot always throw parts of the input away.

If we accept, therefore, that more serious approaches to compound noun processing are required, what does this imply?

Processing compound nouns has implications for system capacity, and for system structure.

The essential problem about compound nouns is that interpreting them requires inference, and specifically pragmatic inference, in an

unpredictable way. Various attempts have been made to characterise compound nouns in terms of general semantic relations, for example PRODUCT, CAUSE, RESEMBLANCE[1,2]. But many compounds cannot be characterised in this way[3], and the particular relations underlying those that can have still to be discovered, for which inference may be required. Moreover, while more than one alternative interpretation of a compound may be tolerated, others may have to be rejected. Equally, though inexplicit representations may serve for some purposes, explicit meaning representations may be required even for apparently text- rather than knowledge-oriented tasks, like translation as opposed to question-answering, and to provide these inference may be necessary.

An illustration

To emphasise these points, consider "border plants" in

"These border plants have to be replaced"

("These plants in the border have to be replaced")

and in

"These border plants are expensive".

("These plants for the border are expensive").

We can say that the general semantic relations underlying "border plants" in these two sentences are LOCATION and PURPOSE respectively. But even if "border plant" figured in the lexicon with these two meanings, we might have to use pragmatic inference to establish which of the two meanings applied in each sentence; and as it is unlikely to appear in the lexicon, we will very probably have to use inference. That is, we cannot assume that we will have standard semantic patterns which will collocate 'plant' and 'replace' for LOCATION an 'plant' and expensive' for PURPOSE. Nor can we assume that our pragmatic knowledge base will contain low-level, directly-applicable facts about border plants needing replacement, etc. We should expect to have to make inferences from more general facts about plants, or even living things, deteriorating or getting old or dying, about ornamental objects typically being items of purchase, and so on.

Notice, moreover, that we have no reason to suppose that domain-specific knowledge structures will help us in any straightforward way here: both sentences could naturally occur in a 'Gardening' frame. We will have to rely at least on techniques for exploiting several frames

concurrently, and almost certainly on applying local textual context information as well.

The refined inference operations needed to sort out the two interpretations of "border plants" just given are equally needed if we accept that "border plants" in the first sentence can mean either 'plants located in the border' or 'plants belonging to the border'. They will also be needed for a text-oriented task like translation, to select appropriate output prepositions in French, say, as much as for a knowledge-oriented task for which referents for "these border plants" have to be found.

If we extend the example to consider e.g. "perennial border plants", or "park border plants" (and, further, "blue border plants"), we have a bracketing problem; and as "park border plants" can be bracketed either as "park (border plants)" - those flowers I saw yesterday - or as "(park border) plants" - those favourites of municipal gardeners, it is highly likely that inference would be needed to establish the correct reading. Equally, as soon as we consider the other senses of the words involved, it is not clear that yet more inference will not be required. We could presumably eliminate 'plant = factory, installation' with more or less standard semantics; but distinguishing 'border = bed' from 'border = edge' is more of a challenge. (What, moreover, would standard semantics do to "The IRA have done so much bombing recently all the border plants need replacing"?)

What the example suggests, therefore, is that while syntactic, semantic and pragmatic information can contribute to the interpretation of compound nouns, their relative contribution for any given compound is unpredictable. Or, to put the point more strongly, the problem for a natural language processor is that it is possible that rather little effective work can be done on an input noun string unless pragmatic inference is invoked. What does this imply for the design of a natural language interpreter?

Compounds and system architectures

First, it is clear that there are problems about the conventional natural language program in which the contents of clearly-demarcated information boxes labelled syntax, semantics, and pragmatics are applied

in successive processing steps. At the high level relevant here it does not matter whether the sequence of steps is followed only for complete input sentences, or for within-sentence constituents, though the detailed consequences of the choice made on this will differ. The essential problem for sequential processing is that at each stage before the pragmatic one compound nouns may be handed over in a very poorly analysed state. Thus syntax may not be able to do much more than label the rightmost element as the head of the compound. For richer complex nominals, some bracketing constraints on the way adjectives and nouns can be grouped may apply, but noun strings as such can generally be bracketed any way, so the syntactic processor is obliged to hand over, implicitly or explicitly, all the alternatives. Moreover if the word forms involved can have other syntactic functions, for example as verbs, we can get yet more options for the semantic processor to consider.

A semantic processor using any kind of pattern matching in turn has the problem that the location of the units to be related in matching is uncertain. Thus if it is not clear, given a multiple noun string, what its constituent groups and their heads are, there is more complex pattern matching to do. The result is likely to be many alternative semantic analyses which are handed forward for weeding to the pragmatics component. These semantic interpretations may of course have been obtained by semantic inference rather than by simple pattern matching. Further, since we cannot expect even semantic inference to select or reach interpretations for all compound nouns, we are likely to be left with non-interpretations as much as with alternative interpretations for the pragmatic component to sort out.

As long as we are not concerned with psychological modelling, the fact that alternative analyses are carried forward may not matter in principle; but it is likely to be very inconvenient in practice, in the limit so inconvenient as to undermine the idea of effective processing on which the staged program is based. There is perhaps some difficulty too at the level of principle about reconciling compound interpretation as selection with compound interpretation as construction of a specific meaning relationship. But even if this is not an issue of principle, there is certainly a practical problem in managing the two.

The natural reaction to these practical difficulties is to modify the rigidly sequential system architecture, to allow for more flexible

processing in which different types of information and procedure can be called at more than one point in the overall attack on the input sentence. A variant of this scheme is that advocated by Charniak[4], where word-based network processing is carried out in parallel with the normal processing, and is allowed to feed its results into any ordinary processing stage. The kind of network-based compound noun resolution apparently developed by McDonald[5] would seem to fit in here.

The advantage of an approach like this is that sentence processing is orderly, indeed the assumption underlying it is that sentence processing <u>is</u> basically orderly. However if enough interaction between, and cycling over, process stages is allowed, it is not clear that the whole notion of processing steps has any meaning. It would be better then to face this fact and go for the radical alternative of non-sequential, and hence word-driven, processing. This looks very attractive as a solution to the compound noun problem: we let all kinds of process, syntactic, semantic, and pragmatic, do their bit, as the individual input words stimulate them. Logically, processing can be parallel, so pragmatic inference can be brought to bear on compound noun string interpretation just as soon as syntactic processing.

Unfortunately, things are not so simple. It is not clear how far word-driven interpretation can capture those gains in applying patterns and rules which come from explicitly working with higher-level constituents. A word-driven interpreter which works implicitly with constituents is obviously subject, if only in a variable, specific way, to those 'staging binds' which occur globally in the modular system. However a word-driven interpreter of a purer sort is also likely to be subject to staging binds. Any interpreter at any time is going to have to pursue one processing path rather than another, simply because life is too exigent even in principle for full parallelism, and can therefore find itself, for compound nouns at least, in a situation where some other processor has not supplied the information a given processor needs.

The difference between the two interpreter architectures, in other words, is not that staging binds will occur in one case and not the other, but that they will occur systematically in one case and unsystematically in the other.

Implications for parsing

These architecture ramifications of compound noun processing point up the limitations of conventional views of parsing and its importance. Conventionally, parsing means syntactic processing; and conventionally it is used to drive everything else. The real problem with compound nouns (and of course even more with complex nominals) is in interfacing the different interpretive processes within the system as a whole. Syntax can do something with compound nouns, and their sentence environments, but not much. The real interest in dealing with compounds is in the rest of the interpreter: how to apply semantic and pragmatic procedures to them. Applying semantics and pragmatics in relation to weakly-informative syntax depends on this.

Compound nouns show that either system model - sequential or parallel - presents problems; but they equally show that concentrating on sentence parsing in its own right is of limited utility. It is the entire complex system, with syntactic processing in a subsidiary role, which counts: we must start with a view of what text interpretation is, and hence of the system as a whole, before we think about how syntactic parsing contributes to interpretation, and so about how a parser should be constructed.

This paper was stimulated by discussions with John Tait.

1. Warren, B. Semantic patterns of noun-noun compounds, Gothenburg Studies in English 41, Goteborg: Acta Universitatis Gothenburgensis, 1978.

2. Levi, J.N. The syntax and semantics of complex nominals, New York: Academic Press, 1978.

3. Downing, P. 'On the creation and use of English compound nouns', Language, 53, 1977, 810-842.

4. Charniak, E. Presentation at IJCAI-81.

5. McDonald, D.B. 'Compound: a program that understands noun compounds', Proceedings of the Seventh International Joint Conference on Artificial Intelligence, 1981, 1061.

Recognising Conjunctions within the ATN Framework

B.K. Boguraev
University of Cambridge Computer Laboratory,
Corn Exchange St., Cambridge CB2 3QG.
England.

1. Abstract

ATN grammars have been widely used for natural language parsing. However, providing a wide and comprehensive coverage of the varying conjunction constructions only through the grammar specification imposes a heavy burden on the grammar writer and produces an extremely bulky grammar. More importantly, attempts to provide a comprehensive set of rules for recognising coordinate constructions using the ATN formalism are destined to fail because of the constraints it places on moving between levels of computation during parsing. The paper presents an alternative approach to the conjunction problem: this is to extend the specification of the ATN interpreter by setting up a set of demons which take over control on encountering a conjunction and interleave their operation with the normal ATN state transition sequence.

This paper presents some ideas about extending the ATN mechanism to begin to deal with conjunctions. The emphasis is on recognising coordinate constructions keyed by "and" only (the assumption being that other conjoined constructs will lend themselves to similar analysis); no attempt will be made to discuss representation problems or the possible structure of a "Conj" semantic specialist. As a result, the proposed mechanism will be as happy with the straightforward "hungry cats and dogs", as with the semantically different cases of symmetrical predicates, as in "Bill and John look alike", or asymmetric "and", as in "The Lone Ranger mounted his horse and rode into the sunset", etc.

2. Conjunctions and ATNs

The ATN formalism, as presented in [Woods70], although quite powerful in expressive power with regard to natural language grammars, faces serious problems when it comes to capturing the various coordinate constructions. To see why this is so, consider the following (sketchy) ATN grammar for very simple noun phrases:

```
Np => (det) (adj)* noun (pp)*

(Np/
   (Cat det ... (to Np/det))
   (Jump Np/det ...))
(Np/det
   (Cat adj ... (to Np/det))
   (Cat noun ... (to Np/Pop)))
(Np/Pop
   (Push pp/ ... (to Np/Pop))
   (Pop (Npbuild) ...))
```

Fig. 1

In order to make explicit all the coordinate Np/ constructions which can be derived from this simple Np/ specification, we will need to add five new arcs to the network:

```
(Np/ ...)
(Np/det
   (Cat adj ...)
   (Cat noun ...)
   (Wrd and (to Np/))      :"the hungry and the lazy cats"
   (Wrd and (to Np/det)))  :"the hungry and lazy cats"
(Np/Pop
   (Push pp/ ...)
   (Wrd and (to Np/))      :"the cats and the dogs"
   (Wrd and (to Np/det))   :"the hungry cats and dogs"
   (Wrd and (to Np/Pop))   :"the cats with whiskers and with long tails"
   (Pop (Npbuild) ...))
```

Fig. 2

Generally, one will need Wrd and arcs to take the ATN interpreter from just about every state in the network back to almost each preceding state on the same level, thus introducing large overheads in terms of additional arcs and complicated tests. This is clearly an undesirable state of affairs. To make things worse, even an explicit extension of the grammar along the lines just indicated will not in fact handle certain types of coordinate construction: gapping, i.e. certain cases of verb or object deletion, as in "Bill designed, and Joe wrote the program", or reduced

conjunctions where the conjoined fragments are not constituents of the grammar, as in "I bumped into and injured a friend".

3. Recognising coordinate constructions by extending the ATN grammar specification

It is possible to adopt an alternative approach to extending the grammar: Blackwell proposes a single Wrd and arc taking the interpreter from the final to the initial state of a computation, ready to analyse the second argument of a coordinate construction on a second pass through the network [Blackwell81]. Clearly this is not a straightforward process, since it may require that a computation be pended and holes be filled. The processing at any level may be interrupted by "and", and after the coordinate structure has been analysed, the deletions in either of the conjoined constituents will have to be undone. Blackwell's approach is based on implementing two constraints in the ATN formalism: the Directionality constraint and the Peripherality constraint [vanOirsouw80]. These help to decide when, which registers, and in which direction, to copy depending on whether the coordinate construction has been derived through forward or backward deletion. Blackwell's proposal is more elegant than the brute force one; but it pays the price of substantial overheads in duplicating almost all registers in all network levels. It also requires the introduction of unnatural Jump arcs in order to impose the Directionality constraint for backward deletion: this, in effect, pends the processing of a constituent and starts a parallel computation for a similar phrase. Blackwell's approach, in other words, bends the ATN formalism to cope with procedural programming constraints, which is, to say the least, an unnatural application for a grammar writing tool of its kind.

Even so, Blackwell's strategy cannot be extended to cope with reduced conjunctions. Indeed, it seems that this class of coordinate structure simply cannot be represented by the ATN formalism, however bent. This is because an ATN provides only limited cross-level process communication provided by the Sendr / Liftr actions, and the constraint that the only exit from a current computation level is through the Pop arc. The problem

here is that the recognition of conjoined fragments of different type requires pending the current level computation, and initiating or restarting another one through a different subnetwork, on a different level! There are no such facilities in the ATN formalism.

4. Recognising coordinate constructions by extending the ATN interpreter specification

All this suggests that what may instead be required is some extension of the way the ATN interpreter works. This is very much what Woods did in implementing his SYSCONJ facility for conjunctions [Woods73]. On encountering "and", the interpreter starts working in special mode. It pends its current processing and tries to branch from some randomly selected preceding point of the computation, to parse the text following the "and". This text is parsed until a point in the input string is reached from which both the resumed and the suspended computations can share the subsequent text. Woods' approach is specially designed to analyse reduced conjunctions, where identical substrings on the left and the right of two conjoined constituents have been factored out, to leave reduced constructs as the two text arguments of "and": e.g. "(I) bumped into (a friend) and (I) injured (a friend)" has the text form "I bumped into and injured a friend".

Analysing reduced conjunctions is the distinctive advantage of this method, but as a general approach it is very costly, because treating all coordinate structures (apart from gapping ones which are not dealt with at all) makes for nasty combinatorial explosions. These are compounded from the alternative choices of past computation to resume, of point at which to resume it, and of time at which to restart the suspended process. Woods is presumably forced to sacrifice processing efficiency because the environment in which his LUNAR parser operates requires a deep transformational analysis of coordinate constructions in order to reconstruct the deletion process which has operated on the conjoined underlying sentences. However, it has been argued elsewhere [Boguraev79] that if ATN syntactic constituent analysis is coupled with semantic constituent assembly within the framework of a passive parser, we can

leave it to later semantic specialist routines to construct a meaning representation from the analysed syntactic components. Specifically if we do not expect the syntactic ATN to deliver a complete explicit (Conj S1 S2) analysis after undoing all the deletions, reductions and transformations, and are willing instead to accept conjoined constituents of any type at any level, because we assume that the system's semantic component will know what to do with an Np with (Conj adj1 adj2), or a Vp with (Conj Np1 Np2), we can avoid many of the overheads and problems discussed.

5. Dynamic arc construction and evaluation

My approach is based on the principle that only categorially identical constituents can be conjoined [Radford81]. (The term category here covers both lexical and phrasal categories). One of the basic problems with coding conjunctions into an ATN is that the ATN is a top-down expectational device, whereas "and" is a bottom-up marker, i.e. predicting where "and" might occur in the input string is a lost cause, because it may occur just about anywhere (see fig.2). However, once it has been detected, we can set up some expectations as to what might follow it. In fact, what could follow it is a constituent categorially identical to the one currently being processed, or just recognised. Indeed, if we look back at fig.2, we will notice that the common feature of the explicit Wrd and arcs is that within a level, all of them return to the beginning of a constituent categorially identical to the one just analysed. Thus, assuming that our interpreter is capable of keeping a history of the parsing process [1] , we can create a demon to be woken up when "and" is encountered. This demon will suspend normal processing, inspect the current context and recent history, and use the information these supply to dynamically construct a new ATN arc which seeks to recognise a constituent categorially similar to the one just completed. This of course means that we need either to construct and attempt the transition of a Cat

---

[1] in terms of the chain of levels of computations (processes) currently active, chains of state transitions through a level, and the actions associated with some important transitions

arc looking for a word belonging to a specific lexical category, or, failing that, to construct and evaluate a Push arc, whose argument can be deduced from the history of the analysis process so far. Clearly, if this fails, we will need to construct another arc, pushing again to a constituent from which the current level of computation was initiated, and so on.

The demon which constructs the arc searching for the conjoined constituent embodies the two constraints mentioned above, namely Directionality and Peripherality, i.e. it decides which registers to initiate by copying existing structures, and which structures to duplicate at the end of a successful dynamic arc transition, at the same time making sure that only well-formed partial structures are copied across. This mechanism is capable of dealing both with forward and backward deletion in coordinate constructions, as well as combinations of these:

* forward deletion:
  "the tall houses with spires and + + + with gables"

* backward deletion:
  "tall + + and austere houses with spires"

* forward and backward deletion:
  "the tall + + and + austere houses down the road".

Note that the dynamic arc construction is carried out until a successful transition of the arc occurs. Thus for the phrase "spaghetti with red sauce and wine", the extended interpreter on its single scan through the network will identify only "sauce and wine" as a conjoined group; the subsequently invoked semantic specialist(s) have to be relied on to get all the semantically alternative readings for the whole phrase. Thus, although the semantic components may have a lot of work to do, we have only a single pass through the text to do the syntactic work.

Reduced conjunctions can be dealt with by an extension of this strategy through the provision of a second demon: this waits for the computation initiated by the dynamic arc evaluation to initiate a search for the constituent expected at the point of the first demon activation (which is easy, because the ATN is a powerful predictive mechanism). On activation the second demon merges and builds a (Conj const1 const2) structure, and

normal ATN processing is then resumed. Finally, gapping, which is a very different phenomenon, has to be considered separately, but it can be dealt with very easily by hard-wiring into the grammar.

6. Conclusions

The paper presents a hybrid method for recognition of coordinate constructions by an ATN parser, based both on (internally) extending the ATN interpreter specification and (externally) extending the ATN grammar. This approach seems to be superior to others proposed in several respects:

1. It allows for more natural extension, compared with [Woods73], of the way the ATN interpreter works in that there is no essential change of its specified mode of behaviour; only a slight interrupt to construct a new data structure, which is external to the grammar, but still totally compatible with the interpreter's view of the world.

2. The decision about which arc to construct for searching for the second component of the coordinate structure is well defined in terms of the processing history of the interpreter. This means that, unlike Woods, it is not necessary to select a past parser configuration to resume at random, which reduces the danger of combinatorial explosion. The desired symmetry between the two constituents can moreover be imposed quite naturally by the actions on the arc.

3. Since any formal grammar of English, together with the Directionality and Peripherality constraints implicitly describes the classes of coordinate structures acceptable to the grammar, there is no need for the grammar writer to worry at all about explicitly specifying the syntax of conjunctions. This, in contrast to Blackwell's approach, clearly saves both man and machine effort.

7. References

[Blackwell81]
Blackwell, S.A. "Processing Conjunctions in an ATN Parser". Unpublished M.Phil. Dissertation, University of Cambridge, 1981.

[Boguraev79]
Boguraev, B.K. "Automatic Resolution of Linguistic Ambiguities". Technical Report No.11, University of Cambridge Computer Laboratory, Cambridge, 1979.

[Radford81]
Radford, A. "Transformational Syntax". Cambridge University Press, Cambridge, 1981.

[vanOirsouw80]
van Oirsouw, R.R. "Deletion Processes in Coordinate Structures in English". Ph.D. Thesis, University of Cambridge, 1980.

[Woods70]
Woods, W. "Transition Network Grammars for Natural Language Analysis". Communications of the ACM, vol.13, 1970.

[Woods73]
"An Experimental Parsing System for Transition Network Grammars". in "Natural Language Processing", Rustin, R. (Ed.), Algorithmic Press, New York, 1973.

Semantic Parsing and Syntactic Constraints

J.I. Tait
University of Cambridge Computer Laboratory,
Corn Exchange St., Cambridge CB2 3QG.
England.

1. Introduction.

This paper considers ways in which semantic parsers can exploit a particular example syntactic constraint which has semantic consequences. By semantic parsers I mean programs which convert natural language sentences into some representation of their meaning. The main conclusion is that there are significant disadvantages in attempting to do semantic parsing without complete syntactic processing of the input.

Two semantic parsers are considered. One is that of Cater ([Cater80], [Cater81]); the other is that of Boguraev ([Boguraev79]). I have selected these two because, from the point of view of this paper, they are reasonable representatives of two styles of parsing. In one style little syntactic analysis is done: processing is based on semantics-driven programs associated with particular words. Cater's program is the representative of this style. Other programs written in this style include [Small80] and [Riesbeck75]. The other style, represented by Boguraev's program, involves complete syntactic processing of input sentences, with semantic processing more or less decoupled from it. Others in this style include [Woods73] and [Winograd71].

Another reason for the choice of these parsers is that I am very familiar with them. I have maintained and developed both of them. Such familiarity is important here because the essence of my argument is that in practice the implementation of the exploitation of the constraint in Cater's parser is beset with technical difficulties. Moreover, these difficulties are products of the style of parsing and not infelicities in Cater's program. Thus the discussion proceeds at a rather technical level.

The syntactic constraint considered in this paper is the well-known no-crossing-of-branches rule [1] and, in particular, its application to prepositional phrase attachment. Consider, for example:

(E1) John saw Mary in the park with the telescope.

The constraint prohibits any reading of this sentence corresponding to:

(E2) John, who was in the park, saw Mary, who had the telescope.

In general the constraint places significant restrictions on the possible readings of sentences with optional post-modifiers, and, as in (E1) it can eliminate possible readings which are perfectly plausible on semantic grounds.

Of course, Phrase Structure Grammars cannot produce analyses which violate the no-crossing-of-branches constraint.

2. The structure of the two parsers.

Boguraev's parser operates by interleaving syntactic and semantic processing. It uses a large ATN grammar for English which invokes so-called semantic specialists whenever a noun phrase or clause can be syntactically terminated. The semantic specialists are handed the partial syntactic analyses in the form of constituents to be assembled. The specialists exploit a semantic primitive system to determine if the constituents they are handed form a semantically plausible whole. If they do, a case-labelled dependency structure is built to represent them. Otherwise the associated syntactic parsing is blocked. The simplicity of this description masks the complexity in Boguraev's system of the ATN-based syntactic processing and the semantic pattern matching. From the point of view of this paper, there is little difference between Boguraev's parser and any other which performs complete syntactic analysis of input sentences whilst producing its meaning representation, regardless of when semantic processing is done. In particular, the discussion below of the

---

[1] See, for example, [Radford81].

exploitation by Boguraev's parser would, in outline, apply to a system which performed complete syntactic analysis of an input sentence and only afterwards did any semantic processing, like [Woods73].

Cater's parser operates by breaking the input sentence into an ordered linear sequence of constituent units, like basic noun phrases, prepositional phrases and verb groups, and then performing semantic processing of these constituents using an expectation-based mechanism not dissimilar to [Riesbeck75]. That is, the fundamental operation of the semantic processing is the application of rules of the form: if the current word is X then it is likely that either constituent A, or constituent B, or constituent C, and so on, follows. This is implemented by attaching coroutines, called requests, to words in the lexicon. As a constituent enters the semantic processor any requests attached to words in the constituent are activated. In general a word-activated request loads other requests which examine the input constituent string for constituents which are expected to follow the original word whose occurrence caused the activation of the first request. For example, one of the requests associated with "give" loads two pairs of requests: one pair deals with two noun phrases following the verb phrase; the other pair will operate if the verb phrase is followed by a single noun phrase and then a prepositional phrase whose preposition is "to". Note that the semantic processing runs through the sequence of constituents in a primarily left to right direction. The requests also build semantic representations of the constituents they analyse.

If attention is restricted to one reading of part of a sentence, at any point in the linear sequence of constituents there will be one active coroutine with a pointer to a particular place in the constituent string. The coroutines which analysed the parts of the sentence to the left of the active coroutine's pointer will have handed it a partially built meaning representation and a set of so-called registers into which arbitrary values can be placed. These registers form the primary means by which coroutines intercommunicate. They are used for such diverse things as passing forward the syntactic subject of a sentence until a main verb is found, passing around temporal information derived from verb tense analysis, and recording constraints on the way partially built structures

may be modified by subsequent processing. The parser produces, as its output, representations in a development of Conceptual Dependency Theory [Schank75].

From the point of view of this paper Cater's parser is very similar to those of [Small80] and [Riesbeck75], though as neither has a constituent analyser, more work is thrown onto their word-activated coroutine mechanism. Their coroutine intercommunication mechanisms also differ from Cater's.

In summary, the essential difference between the two parsers is that Boguraev completely processes a sentence syntactically, so that any syntactic information required by the semantic specialists may be given to them, whereas Cater only uses syntactic information to group input words into small constituents.

3. Incorporating the constraint into Boguraev's parser.

Boguraev's parser explicitly uses the no-crossing-of-branches constraint to restrict the readings which may be found for sentences like (E1). Two implementations have been used. The original version of the program, that described in [Boguraev79], used a rather complicated technique, which will not be described here. More recently I have completed an implementation for Boguraev's parser which exploits the constraint in an entirely straightforward way.

In effect, the implementation operates by ensuring that only syntactic analyses which could be generated by a Phrase Structure Grammar are discovered by the ATN mechanism. Both the noun phrase and verb phrase subnetworks consider, as an alternative, processing post-modifying prepositional phrases syntactically before calling their semantic specialists. For example, in (E1) for the noun phrase containing "Mary", the syntactic constituent analyses produced are "Mary", "Mary in the park" and "Mary (in the park) (with the telescope)"; and the sentence-level clause specialist is handed structures corresponding to "John saw", "John saw with the telescope", and "John saw (in the park) (with the telescope)".

The semantic specialists simply check to see if they have been handed a constituent analysis which contains such optional post-modifiers, and if they have, they construct dependency structures for all the semantically plausible ways the prepositional phrases may modify the head noun of the noun phrase or main verb of the clause.

4. Incorporating the constraint into Cater's parser.

Cater's parser is a suitable vehicle for this discussion because it is one of the few in its class which is designed to take account of the possibility of structural, rather than lexical, ambiguity in its input. However it has never actually been used to produce multiple meaning representations for single sentences, and does not exploit the constraint. Therefore this section is rather more hypothetical than the preceding one.

In Cater's system optional post-modifiers are handled by attaching requests to the relevant prepositions. Disregarding the constraint for the moment, if the present implementation were extended to produce all the semantically plausible readings of (E1) (a fairly straightforward matter) it would proceed as follows. A coroutine would be started by the word "in". It would have been handed the representation built for "John saw Mary", and would construct a suitable representation for the prepositional phrase. It would then generate two daughter coroutines. In one of them the structure would represent "John, who was in the park, saw Mary"; in the other the structure would represent "John saw Mary, who was in the park". Each of these coroutines would spawn a daughter when "with" was read; and then, as with "in", yet more coroutines would be spawned by these third generation prepositional phrase attachment coroutines. In the structures in the fourth generation, representing the set of possible readings of the sentence as a whole, the representation for "with a telescope" would be attached to all the semantically plausible points. Six coroutines would therefore be generated, including one in which the structure would reflect the constraint violating reading (E2).

It is possible to see how the no-crossing-of-branches constraint might be introduced into Cater's parser. The main difficulty to be overcome is

that the semantic representation cannot reflect the left-to-right order of constituents, nor even which parts of the representation were derived from prepositional phrases. This is not purely a product of the nature of the representation language, however. The correct ordering of cases in Boguraev's current implementation is guaranteed by complete syntactic processing of a sentence. Cater's system has as a central feature that "syntactic processing should be used ... to locate groups of words which should be treated as a whole" ([Cater81] page 107) and nothing else. (This is in fact more use of syntactic information that other proponents of this style of parsing, for example [Riesbeck75], allow.)

It seems that the only way to implement the no-crossing-of-branches constraint within the current framework of Cater's parser would be as follows. Whenever an element was added to the Conceptual Dependency structure being built for the sentence, a pointer [2] to that element would be placed in a register. This register would be maintained in such a way that the order of the pointers in it always corresponded to the the textual order of the constituents which gave rise to the objects pointed to. Whenever a prepositional phrase was taken from the input, in addition to any other necessary activities, a link would be placed in the pointer register connecting the prepositional phrase and the object it modified. Requests activated by the use of prepositions as post-modifiers would check, when updating this register, whether they were about to place a link of their own which would break a link created by a previously attached prepositional phrase. No daughter coroutines would be created for points of attachment with this property.

The maintenance of such a bracketed list of constituents (which this register would effectively be) looks very much like syntactic analysis. It is constructed and used, however, by a mechanism primarily intended to do semantic analysis. Because of this the implementation of the constraint looks clumsy and unnatural compared with one based on a mechanism intended for syntactic analysis. In addition, if this process is taken to be

---

[2] Pointers are used to overcome the difficulty of relating the very deep Conceptual Dependency structures to the order of the surface text.

syntactic analysis it violates Cater's stated principle that the analyser should not perform complete syntactic analysis of whole sentences.

Riesbeck and Small would need to use equally contorted simulations of syntactic analysis if they were to deal with this sort of structural ambiguity and wished to use the no-crossing-of-branches constraint to ensure that illegitimate readings like (E2) were not found for sentences like (E1). The details would be different from that proposed for Cater's parser, but only because of the differences in their coroutine intercommunication mechanisms.

5. Conclusions

The conclusion that must be drawn from these implementations of the no-crossing-of-branches constraint is that, if a semantic parser operates without a complete syntactic parse of its input, it is difficult, if not impossible, to prevent it finding readings which do not in fact exist. With a complete syntactic parse it is easy to block the possible discovery of at least one class of such spurious readings.

Parsers which rely heavily on the use of semantic and even pragmatic, rather than syntactic, information do have their virtues. Thus at present it appears easier to construct robust parsers, capable of handling wide varieties of everyday texts, by emphasizing the use of such information. However, it seems plain that if such systems ignore syntactic analysis they will consider entirely illegitimate interpretations of their input.

Acknowledgements: I would like to thank Dr. K. Sparck Jones and Dr. B. K. Boguraev for their advice and assistance during the preparation of this paper.

References

[Boguraev79]
Boguraev, B.K. "Automatic Resolution of Linguistic Ambiguities". Technical Report No. 11. University of Cambridge Computer Laboratory, Cambridge. 1979.

[Cater80]
Cater, A.W.S. "Analysing English Text: A Non-deterministic Approach with Limited Memory" AISB-80 Conference Proceedings. Society for the Study of Artificial Intelligence and the Simulation of Behaviour. July 1980.

[Cater81]
Cater, A.W.S. "Analysis and Inference for English" Unpublished Ph.D. Thesis. University of Cambridge. 1981.

[Radford81]
Radford, A. "Transformational Syntax" Cambridge University Press, Cambridge. 1981.

[Riesbeck75]
Riesbeck, C.K. "Conceptual Analysis" in [Schank75].

[Schank75]
Schank, R.C. "Conceptual Information Processing" North-Holland, Amsterdam. 1975.

[Small80]
Small, S. "Word Expert Parsing: A Theory of Distributed Word-based Natural Language Understanding" TR-954, Department of Computer Science, University of Maryland, College Park, Maryland. 1980.

[Winograd72]
Winograd, T. "Understanding Natural Language" Edinburgh University Press, Edinburgh. 1972.

[Woods73]
Woods, W. "An Experimental Parsing System for Transition Network Grammars", in Natural Language Processing, Rustin, R. (Ed.) Algorithmic Press, New York. 1973.