Interactive semantic analysis of Clause-Level Relationships

KEN BARKER & STAN SZPAKOWICZ
Department of Computer Science, University of Ottawa
Ottawa, Ontario, Canada K1N 6N5
email: {kbarker, szpak}@csi.uottawa.ca
phone: +1 613 562 5826 fax: +1 613 562 5187

Abstract

Natural Language Processing (NLP) systems usually require large amounts of pre-coded domain knowledge to perform semantic analysis automatically. Until repositories of such background knowledge are widely available, these systems may not scale up to non-trivial applications of NLP.

This paper describes the design and implementation of a system that uses surface-syntactic information to interpret interactively semantic relationships between clauses. English technical texts are analyzed by a domain-independent parser that produces detailed parse trees of the input. The system then examines clausal connectives and syntactic verb phrase features to determine what kinds of semantic relationships exist between clauses. The results of this activity are used in a large Knowledge Acquisition system that, by design, requires little *a priori* semantic knowledge. We present a set of semantic labels appropriate to syntactically connected clauses (Clause-Level Relationships) and a description of the theory behind assigning these labels to particular inputs (Clause-Level Relationship Analysis). We also discuss elements of the implementation of the Clause-Level Relationship Analyzer and look at its performance.

1 Introduction

The objective of the TANKA project is to build a conceptual model of a technical domain by processing English text that describes the domain. A guiding principle of the project is to use as little *a priori* semantic knowledge as possible. In the absence of such knowledge, processing is driven by the surface syntax and assisted by a user. TANKA's text analysis system starts with an unsimplified text and applies syntactic knowledge and surface-level lexical knowledge to build a conceptual model of the domain. The help of a user is enlisted to verify the analyses made by the system. Texts are minimally pre-edited to remove non-textual items (such as control characters, illustrations, etc.)

The TANKA system consists of three phases: syntactic analysis, semantic analysis and conceptual network building.

Syntactic analysis is performed by DIPETT (Domain Independent Parser of English Technical Texts). DIPETT is a broad-coverage DCG parser. Its grammar does not adhere to a particular existing linguistic theory of grammar. Rather, its rules are based primarily on Quirk *et al.* (1985). See Copeck *et al.* (1992) and Delisle (1994) for details on DIPETT.

The semantic analysis module, called HAIKU, consists of three levels of semantic processing: Clause-Level Relationship Analysis, Case Analysis and Noun-Modifier Relationship Analysis.

Clause-Level Relationships (CLRs) are the semantic relationships between acts, events or states represented syntactically by syndetically connected finite clauses. CLR Analysis attempts to assign a semantic label to the relationship expressed in the connection of these clauses.

Cases represent semantic relationships between the main verb within a clause and its syntactic arguments (subject, objects, prepositional phrases and adverbials). For each clause in the input, HAIKU's Case Analyzer interactively assigns Case labels to the syntactic arguments based on syntactic and lexical clues as well as Case patterns memorized from previous processing.

The Case Analyzer is described in detail in Delisle *et al.* (1994).

Noun-Modifier Relationships (NMRs) represent semantic relationships between the head noun of a noun phrase and its modifiers (adjectives, nouns, relative clauses, etc.). Research on NMRs and linguistic foundations of NMR Analysis is in progress.

The division of semantic relationships into these three levels follows from their syntactic analysis at different levels, but we recognize the fact that a single semantic concept may be realized syntactically at any one of the levels. For example, the following sentence and noun phrase represent the same basic concepts and relationships between them:

A man was murdered yesterday with a handgun because his lover's jealous husband returned home early.

the murder of a man yesterday with a handgun because of the early return of his lover's jealous husband

Consequently, there is some overlap in the three sets of relationships.

The structures resulting from semantic analysis will be used by a module called the Network Fragment Builder to construct conceptual network fragments—clusters of concepts. These fragments will be added to a growing network intended to be a representation of the domain. Details of the Conceptual Network formalism can be found in Yang & Szpakowicz (1991a, 1991b, 1994).

2 Clause-Level Relationships

The CLR Analyzer recognizes a small set of general semantic relationships using a list of the lexical items that mark them (coordinators, correlatives, subordinators). In this paper, these lexical items are referred to as *CLR Markers* (in a semantic context) or *clausal connectives* (in a syntactic context). The CLRs themselves were chosen after an exhaustive study of the various meanings of the connectives. Barker (1994) contains a more detailed account of the construction of the CLR set.

Earlier versions of the set of CLRs were refined by comparing them to lists of semantic relationships previously presented in traditional and computational linguistics—specifically, in Discourse Analysis. Traditional "Discourse Relations" deal not only with the semantic relationships between the events or acts represented by clauses, but also with the rhetorical functions of the clauses themselves. The CLRs presented here (see Table 1) only deal with semantic relationships.

Causal
Causation (caus)
Enablement (enab)
Entailment (entl)
Prevention (prev)
Detraction (detr)
Temporal
Temporal Co-occurrence (ctmp)
Temporal Precedence (prec)
Conjunctive
Conjunction (conj)
Disjunction (disj)

Table 1: Clause-Level Relationships

2.1 The CLR Marker Dictionary

The CLR Marker Dictionary consists of entries enumerating the CLRs that can be associated with each marker. These CLR Marker→CLR mappings were determined by studying all of the usages of each connective to learn what semantic relationships they represent.

Inspection of the connective usages also uncovered a result useful to CLR Analysis: the identification of a consistent "direction" for each connective. For most of our CLRs, the order of the arguments is relevant. The exceptions are *Conjunction*, *Disjunction* and *Temporal Co-occurrence*. For example, for *Causation*, one clause is the antecedent and the other is the consequent. With some connectives, the clause introduced by the connective is the antecedent (e.g. 'The file will print *if the program works*') whereas with other connectives the clause introduced by the connective is the consequent (e.g. 'The program works *so the file printed*'). This correspondence between the syntactic arguments to the connective and the semantic arguments to the CLR is consistent among the usages of the connective. Each connective's direction is stored in the CLR Marker Dictionary so that CLR Analysis can automatically determine the correct ordering of arguments, given the order of the syntactic arguments.

2.2 CLR Definitions

Our set of Clause-Level Relationships is divided into three groups according to the type of relationship the CLRs represent: *Causal*, *Temporal* and *Conjunctive*. This division corresponds to a kind of ranking: The *Causal* CLRs imply a temporal ordering (a cause temporally precedes its effect) and also imply a *Conjunctive* relationship (*Prevention* implies *Disjunction*; the other *Causal* CLRs imply *Conjunction*). Similarly, the *Temporal* CLRs imply *Conjunction*. In the absence of any other information, this ranking could be used as a default to prefer one CLR over another. For example, by default *Temporal Precedence* may be preferred over *Causation*, since *Causation* implies *Temporal Precedence*. However, the system can afford to be ambitious and prefer the "stronger" *Causation*, since the user can always reject the suggested assignment.

The *Causal* CLRs are all binary. The *Temporal* and *Conjunctive* CLRs can have more than two arguments. However, the only connectives that can mark these n-ary CLRs (n > 2) are the coordinators 'and' and 'or'. Of these two, 'or' unambiguously marks both inclusive and exclusive *Disjunction*, while 'and' can mark *Conjunction*, *Temporal Co-occurrence* or *Temporal Precedence*. The rest of the CLRs present more difficult disambiguation problems. Therefore, the mechanisms described in section 3 apply to binary CLRs.

In the following descriptions, A1 and A2 refer to acts or states denoted by clauses. They correspond to the "first" and "second" CLR (semantic) arguments. The connective appears in uppercase in the examples and where the distinction between A1 and A2 is relevant, A2 appears boldface.

Causal CLRs

The *Causation* relationship represents the situation where A1 makes A2 occur or exist. A1 is sufficient to cause A2 and the occurrence or existence of A1 is required.

The file printed *BECAUSE the program issued a print command.* *(caus)*

The *Enablement* relationship represents the situation where A1 makes A2 possible. A1 is necessary to enable A2 but is not sufficient.

The printer can print *IF the paper tray contains paper.* *(enab)*

For the *Entailment* relationship, if A1 exists or occurs then A2 must also exist or occur. Unlike *Causation*, however, A1 is not known to exist or occur.

The printer will print *IF a print command is issued.* *(entl)*

A *Prevention* relationship exists where A1 is meant to keep A2 from occurring or existing. A1 is sufficient to prevent A2.

The files were not copied *SINCE the hard disk crashed.* *(prev)*

The *Detraction* relationship corresponds to the situation where A1 detracts from A2 but is insufficient to prevent A2 from occurring or existing.

ALTHOUGH the server was very busy, ***the program ran****.* *(detr)*

Temporal CLRs

Temporal Co-occurrence represents the relationship in which A1 and A2 occur or exist at the same time.

One job can run in the background WHILE another job runs in the foreground. *(ctmp)*

Temporal Precedence represents the relationship in which A1 occurs or exists (or begins to occur or exist) before A2.

The file printed BEFORE ***I changed the toner cartridge****.* *(prec)*

Conjunctive CLRs

A *Conjunction* relationship exists between acts or states about which no more can be said than that they both occur or exist.

The computer runs applications AND the printer prints documents. *(conj)*

A *Disjunction* relationship exists between acts or states about which no more can be said than that one or both occur or exist.

The program may terminate OR it may hang indefinitely. *(disj)*

3 CLR Analysis

CLR Analysis begins by finding a connective in an input sentence and consulting the CLR Marker Dictionary to determine which CLRs are marked by it. If the connective can mark more than one CLR, the Analyzer inspects the clausal connective and the verb phrase features of each clause to choose among CLR candidates.

3.1 Polarity of Connectives and Arguments

The polarity of the verb phrase (positive or negative) can be used to distinguish between positive (*Causation*, *Enablement*, *Entailment*) and negative (*Detraction*, *Prevention*) *Causal* CLRs. For example, the difference between *Entailment* and *Prevention* is often the fact that the second semantic argument, or "resultant" (shown boldface) is positive for *Entailment* but negative for *Prevention*:

The document will not print *IF the paper tray is empty.* *(prev)*

The document will print *IF the printer is ready.* *(entl)*

However, with certain connectives verb phrase polarity has the opposite effect in determining the CLR:

The document will print *UNLESS the paper tray is empty.* *(prev)*

The document will not print *UNLESS the printer is ready.* *(enab)*

The "positivity" or "negativity" of a connective will be referred to as the *connective polarity*. This terminology is not meant to suggest that the connective itself is inherently positive or negative. Rather, it reflects the connective's relationship to verb phrase polarity in determining whether the CLR is positive or negative.

The examples also illustrate the need for care in producing CLR structures. For the first sentence, the CLR representation should be *('the paper tray is empty' <prevents> 'the document will print')*, with the negation operator ('not') deleted. The negation should also be removed for the fourth sentence: *('the printer is ready' <enables> 'the document will print')*. To account for this phenomenon, we introduce the following *Polarity-Reversal Rule*:

For Causal CLRs: reverse the polarity of the second CLR argument in the final CLR representation if

a) the CLR is a negative CLR marked by a positive connective, or

b) the CLR is a positive CLR marked by a negative connective

Other negative connectives include: 'although', 'but', 'either-or', 'except', 'only', 'or', 'save that', 'unless', 'until', 'yet'.

Note that polarity is often implicit in one of a clause's elements. Compare 'the program will not succeed' with 'the program will fail' (implicitly negative verb) or 'the program will experience failure' (implicitly negative complement). Polarity can be used in the first example to assist CLR Analysis. The other two examples would require the lexical semantic information that 'fail' and 'failure' are implicitly negative. However, the absence of this information will not result in an erroneous interpretation: 'The program will fail to terminate IF there are bugs' will be interpreted as *('there are bugs' <entails> 'the program will fail to terminate')*. This interpretation is valid, if not as elegant as *('there are bugs' <prevents> 'the program will terminate')*.

3.2 Argument Tense and Modality

In a surface syntactic analysis, tense and modality are often ambiguous. For example, the auxiliary *could* is sometimes used as the past tense of the modal of Ability *can* and is sometimes used as a conditional auxiliary. Similar tense/modality ambiguities exist with *may*, *will* and others (see Lyons (1977) for a linguistic description of the overlap between futurity and modality). These ambiguities suggest that tense and modality be considered together.

Tenses and Modality often hold clues to the semantic relationships between clauses. The difference between many of the *Causal* CLRs is the degree of confidence that the resultant state or event (A2 in section 2.2) will occur. For example, *Entailment* implies a high confidence that the resultant will occur; *Enablement*, a much weaker confidence.

Table 2 shows the modals (and marginal modals) recognized by DIPETT, divided into "stronger" and "weaker" modals. The division is based on research on the English modals, particularly Palmer (1979), Coates (1983) and Quirk *et al.* (1985). Hermerén

(1978) goes further to present multiple levels of modal "strength". However, the definition of the *Causal* relationships as sufficient or insufficient suggests that a binary division of modals is appropriate for disambiguating between CLRs.

Stronger Modals
<no modal>
cannot
dare not
must
need
shall
will
would
Weaker Modals
can
could
may
might
need not
ought
should

Table 2: Strength of the Modals

3.3 Distinguishing the CLRs

For each pair of CLRs (36 such pairs for 9 CLRs), the polarity, tense and modality features were examined to determine how they reflect choice of CLR. The result was a set of "preference rules" for distinguishing between any two CLRs, based on the syntactic features of the clauses. Using these rules, the system will suggest a CLR assignment to the user, who can accept or reject the suggestion.

For some pairs, the syntactic features of the arguments are not consistently different enough to allow disambiguation. These ambiguous pairs have an effect on the outcome of CLR competitions, described in section 4.1.

Here are two examples of the preference rules. Again, resultants are shown boldface in the examples.

Enablement vs. Entailment

The resultant of *Enablement* has weaker modals whereas the *Entailment* resultant contains stronger modals.

IF the power is on, ***the computer can work****.* *(enab)*

IF the program responds, ***the computer must be working****.* *(entl)*

Entailment vs. Prevention

Although *Entailment* and *Prevention* are both sufficient *Causal* CLRs, they differ in polarity. With a positive connective *Entailment* should have a positive resultant and *Prevention* should have a negative resultant. If the connective is negative, *Entailment* should have a negative resultant and *Prevention* should have a positive resultant.

IF the printer has paper, ***the file will print****.* *(entl—pos. conn.)*

IF the printer is out of paper, ***the file will not print****.* *(prev—pos. conn.)*

UNLESS the printer has paper, ***the file will not print****.* *(entl—neg. conn.)*

UNLESS the printer is out of paper, ***the file will print****.* *(prev—neg. conn.)*

4 Implementation

The CLR Analyzer consists of two main modules. The CLR Driver interprets the various syntactic formats of connected clauses output by DIPETT and invokes the CLR Assignment module. Nested clauses, user interaction and CLR structure building are all handled by the CLR Driver.

For nested structures, analysis begins with the innermost nested pairs (or sequences) of clauses and proceeds to the outermost relationships. Consider the following example:

The printer can print if the program issues the print command before the job terminates.

For this sentence DIPETT will produce the following structure:

```
[*statement1*,'if',
   [*statement2*,'before',*statement3*]]
```

where `*statement1*` refers to the subtree for *'the printer can print'*, `*statement2*` refers to the *'the program issues the print command'* and

```
Unit #1: Since Rosa is a supporting person with a net income below $20,000, she can
         claim the child care expenses.

<parse tree not shown>
<anaphora resolution interaction not shown>

HAIKU: Clause-Level Relationship Analysis of current input ...
> There is a Clause-Level Relationship marked by 'since':
       'rosa can claim the child care expenses'
          'since'
       'rosa is a supporting person with a net income below $20000'

CLR competition between prec and caus... caus wins.
CLR competition between prec and prev... tie.
CLR competition between prec and entl... entl wins.
CLR competition between prec and enab... enab wins.
CLR competition between caus and prev... caus wins.
CLR competition between caus and entl... entl wins.
CLR competition between caus and enab... enab wins.
CLR competition between prev and entl... entl wins.
CLR competition between prev and enab... enab wins.
CLR competition between entl and enab... enab wins.

Results (maximum is 8):
   prec   caus   prev   entl   enab
 +------+------+------+------+------+
     1      4      1      6      8

> The CLR Analyzer's best suggestion(s) for this input:
       Enablement (enab)  (1)
> Please enter a number between 1 and 1
  or enter a valid CLR label for this relationship (CR to abort): 1
> Your CLR assignment will be stored as:
       'rosa is a supporting person with a net income below $20000'
          <enables>
       'rosa can claim the child care expenses'
> Do you accept this assignment
  (enter r to reverse the arguments, a to abort) [Y/n/r/a]? y
```

Figure 1: A Typical CLR Analysis Session

`*statement3*` to *'the job terminates'*. The CLR Analyzer will first label the relationship between `*statement2*` and `*statement3*` (this relationship constitutes a composite statement *S*). Next it will label the relationship between `*statement1*` and *S*.

The CLR Assignment module chooses the best CLR to suggest to the user for verification. The CLR Marker Dictionary is consulted for the CLRs marked by the given connective. From these candidates, if this module manages to determine a single best CLR for the relationship, it will be submitted to the user for acceptance or rejection. If decisive disambiguation between the valid CLRs is not possible, the user will be asked to select the most appropriate CLR from the list of candidates for this input.

4.1 CLR Competitions

Each preference rule chooses between a given pair of candidate CLRs. However, if there are more than two candidates, the rules must be applied to individual pairs within the set of candidates. A simplistic approach would be to apply the rules to the first two CLRs to determine the preferred of those two, then apply the rules to the third CLR and the "winner" of the first competition, etc.

For example, suppose there are four candidate CLRs (A, B, C, D) for some input. Suppose further that the rules provide the following preferences for this same input: prefer B over A; prefer B over D; prefer C over B; prefer D over C. Using the simple approach described above, the system would select D as the most appropriate CLR for the input. However, if the

Single CLR Chosen as Winner		*Single CLR with Most Points*		*Multiple CLRs with Most Points*
Accepted	Rejected	Accepted	Rejected	*(ambiguous)*
83	2	11	0	4

Table 3: CLR Competition Test Results

selection algorithm worked backwards through the list starting with the last two candidates, it would select B as the most appropriate CLR. To avoid this inconsistency, the system should hold competitions between *all* pairs of candidate CLRs and choose the CLR with the best preference record. Since it is not always possible for the system to decide between two CLRs, the system must also allow for "ties".

So in the CLR Assignment process, each candidate CLR competes against all other candidates. Each time a candidate is preferred over another candidate, the "winner" collects two points, while the "loser" collects no points. If the rules are unable to prefer one candidate over another, the match is declared a tie and each candidate receives a single point. Once all matches have been played, the CLR with the most points (the "victor") is presented to the user for approval as the most likely CLR appropriate to the given input.

Figure 1 shows the highlights of the CLR Analysis session for a sentence from the 1990 Canadian T4043E Tax Guide.

4.2 Test Results

To check the accuracy of the preference rules and the validity of the competition model, a test was conducted with 100 sentences from the Ontario Building Code (OMH, 1991). The Ontario Building Code is the legal document setting out the regulations for the design and construction of buildings in Ontario; it contains more than 400,000 words. For the test, sentences were chosen containing clauses connected by connectives known to mark two or more CLRs (from the CLR Marker Dictionary). Apart from this restriction, the sentences were chosen randomly from throughout the entire text. For each sentence, DIPETT provided the tense, modality and polarity features of each clause. The CLR Analyzer then held competitions to determine the most appropriate CLR based on these features and the preference rules. The results are summarized in Table 3. The first two columns reflect those sentences for which a single CLR won *all* matches against other CLRs (and the ratio of acceptance to rejection of the chosen CLR by the user). The next two columns correspond to sentences for which a single CLR accumulated more points than any other in the competition, even if it didn't win every match. The fifth column represents competitions with more than one CLR tied for first place.

These results suggest that:

- the CLR Marker Dictionary and preference rules adequately cover the test sentences;
- it is useful to present the CLR with the most points to the user first even if it did not win every match (all 11 of these suggestions were accepted in this test);
- the syntactic features contain enough information for the system to choose a single CLR for most sentences;
- the user is still required to correct poor suggestions (columns 2 & 4) and to disambiguate when the system is unable to do so (column 5).

5 Related Work

Several authors have proposed sets of relationships between clausal (and larger) text elements. Many of these do not distinguish between semantic and rhetorical (or "discourse") functions.

Schank's original *Conceptual Dependency Theory* (Schank, 1975) only defined primitive acts and the relationships between acts and their participants. However, a later refinement to the theory (Schank & Abelson, 1977) introduced *Conceptual Relations* (including *Enable*, *Result*, *Reason* and *Initiate*) to capture the semantic relationships between acts.

Halliday & Hasan (1976) and Van Dijk (1977) both present lists of relations and investigate the distinction between those that are semantic and those that are rhetorical.

Hobbs (1983) presents a taxonomy of *Coherence Relations* that includes *Enablement*, *Cause* and *Contrast* along with several discourse relations. The list of *Rhetorical Relations* in Mann & Thompson (1988) has relations roughly corresponding to all of our CLRs. Numerous other sets of relations have been proposed based on *Coherence Relations* and/or *Rhetorical Relations* (including Hovy (1993), Knott & Dale (1993), Lascarides *et al.* (1992) and Sanders *et al.* (1992) among others). Knott & Dale also give an extensive list of potential "relational cue phrases" (similar to our CLR Markers). Dahlgren (1988) presents a set of *Coherence Relations* that is a synthesis of several others' lists.

Bäcklund (1984) identifies six types of clauses (*Temporal*, *Concessive*, *Conditional*, *Comparative*, *Conditional-Concessive* and *Locative*) defined by their semantic function within the discourse. For each type, connectives that typically introduce clauses of that type are enumerated.

Schiffrin (1987) presents a list of *Discourse Relations* based on a study of the lexical items (Discourse Markers) that signal them. Although these relations are again based on discourse function, Schiffrin's list more closely matches our CLRs and is also divided into *Conjunctive*, *Causal* and *Temporal* relations.

The semantic functions of tense and modality have received treatment in various works in Linguistics. Hermerén (1978) presents a list of several semantic modalities and the particular modal auxiliaries that mark them. He also presents hierarchies of semantic modalities. Palmer (1979) lists semantic modalities as expressed by each of the English modals and identifies two "degrees" of modality: Necessity and Possibility. Coates (1983) makes a similar distinction of "strong" and "weak" between several of the semantic modalities. Quirk *et al.* (1985) gives a mapping between modal auxiliaries and semantic modalities and also makes note of how tense affects modality.

6 Future Work

The techniques described earlier for CLR Analysis choose between candidate CLRs based on the syntactic features of the CLR arguments. However, the arguments may be embedded CLR structures. Since these structures represent combinations of clauses, they have no obvious unique modality or polarity. A CLR competition could be attempted with the embedded CLR argument. Many of the preference rules check the features of only one of the arguments. If the input contains one embedded CLR structure and one clausal argument, a competition may still be possible.

As mentioned in section 3.1, an argument's polarity is often implicit in the semantics of the clause's verb or complement. Since the system does not make use of any lexical semantic knowledge, it cannot use the polarity of an implicitly negative clause.

The modality of a clause is not always conveyed by a modal auxiliary. Adverbs (such as 'possibly' and 'certainly') and modal paraphrases (such as 'it is possible that' and 'it is certain that') are often used to express modality. Again, in the absence of lexical semantics, the CLR Analyzer is unable to interpret these forms as expressing modality. However, it might be tractable to precode this lexical semantic information, since the set of implicitly modal words is probably quite small and closed.

Acknowledgments

This work is supported by the Natural Sciences and Engineering Research Council of Canada. The authors gratefully acknowledge the contributions of several people to the work described in this paper. Sylvain Delisle implemented the first incarnation of HAIKU, laying the foundation for subsequent work in semantic analysis in TANKA. Jean-François Delannoy's close work with the authors and his efforts in translating CLR structures into a logical representation have been a positive force in the development of the theory and the system. We would also like to thank Jean-Pierre Corriveau and Stan Matwin for their careful scrutiny and valuable criticism of an earlier version of this paper.

References

BÄCKLUND, INGEGERD (1984). *Conjunction-Headed Abbreviated Clauses in English.* Stockholm: Almqvist & Wiksell, International.

BARKER, KEN (1994). "Clause-Level Relationship Analysis in the TANKA System". TR-94-07, Department of Computer Science, University of Ottawa.

COATES, JENNIFER (1983). *The Semantics of the Modal Auxiliaries.* London: Croom Helm.

COPECK, TERRY, SYLVAIN DELISLE, & STAN SZPAKOWICZ (1992). "Parsing and Case Analysis

in TANKA". *Proceedings of COLING-92*. Nantes, France, 1008-1012.

DAHLGREN, KATHLEEN (1988). *Naïve Semantics for Natural Language Understanding.* Boston: Kluwer Academic Publishers.

DELISLE, SYLVAIN (1994). "Text Processing without A-Priori Domain Knowledge: Semi-Automatic Linguistic Analysis for Incremental Knowledge Acquisition", Ph.D. thesis, TR-94-02, Department of Computer Science, University of Ottawa.

DELISLE, SYLVAIN, KEN BARKER, TERRY COPECK & STAN SZPAKOWICZ (1995). "Interactive Semantic Analysis of Technical Texts". *Computational Intelligence* (to appear).

HALLIDAY, M.A.K. & RUQAIYA HASAN (1976). *Cohesion in English.* London: Longman.

HERMERÉN, LARS (1978). *On Modality in English: A Study of the Semantics of the Modals.* Lund: CWK Gleerup.

HOBBS, JERRY (1983). "Why is Discourse Coherent?". Fritz Neubauer (ed.). *Coherence in Natural Language Texts.* Hamburg: Buske.

HOVY, EDUARD H. (1993). "Automated Discourse Generation Using Discourse Structure Relations". Fernando C.N. Pereira & Barbara J. Grosz (eds.). *Artificial Intelligence* 63:1-2 (October, 1993), 341-385.

KNOTT, ALISTAIR & ROBERT DALE (1993). "Using Linguistic Phenomena to Motivate a Set of Rhetorical Relations". Human Communication Research Centre, University of Edinburgh.

LASCARIDES, ALEX, NICHOLAS ASHER & JON OBERLANDER (1992). "Inferring Discourse Relations in Context". *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics.* Newark, Delaware, 1-8.

LYONS, JOHN (1977). *Semantics.* Cambridge: Cambridge University Press.

MANN, WILLIAM C. & SANDRA A. THOMPSON (1988). "Rhetorical Structure Theory: Toward a functional theory of text organization". *Text* 8:3 (1988), 243-281.

ONTARIO MINISTRY OF HOUSING (1991). "The Ontario Building Code". Ontario Regulation 413/90. Queen's Printer for Ontario.

PALMER, F. R. (1979). *Modality and the English Modals.* London: Longman.

QUIRK, RANDOLPH, SIDNEY GREENBAUM, GEOFFREY LEECH & JAN SVARTVIK (1985). *A Comprehensive Grammar of the English Language.* London: Longman.

SANDERS, TED J.M., WILBERT P.M. SPOOREN & LEO G.M. NOORDMAN (1992). "Toward a Taxonomy of Coherence Relations". *Discourse Processes* 15 (1992), 1-35.

SCHANK, ROGER C. (1975). *Conceptual Information Processing.* Amsterdam: North-Holland Publishing Company.

SCHANK, ROGER C. & R.P. ABELSON (1977). *Scripts, Plans, Goals and Understanding.* Hillsdale, NJ: Erlbaum.

SCHIFFRIN, DEBORAH (1987). *Discourse Markers.* Cambridge: Cambridge University Press.

VAN DIJK, TEUN A. (1977). *Text and Context: Explorations in the Semantics and Pragmatics of Discourse*. London: Longman.

YANG, LEIXUAN & STAN SZPAKOWICZ (1991). "Inheritance in Conceptual Networks". *Proceedings of the Sixth International Symposium on Methodologies for Intelligent Systems*. Charlotte, NC, 191-202.

YANG, LEIXUAN & STAN SZPAKOWICZ (1991). "Planning in Conceptual Networks". F. Dehne, F. Fiala and W.W. Koczkodaj (eds.). *Advances in Computing and Information - ICCI '91. Lecture Notes in Computer Science* 497, Springer-Verlag, 669-671.

YANG, LEIXUAN & STAN SZPAKOWICZ (1994). "Path-Finding in Networks" *Journal of Computing and Information. Special Issue: Proceedings of the Sixth International Conference on Computing and Information* 1:1, 1003-1018.