

<b>5</b>	<b>Future Work</b>	<b>135</b>
<b>5.1</b>	<b>Clause Level Relationship Analysis</b> .....	<b>135</b>
5.1.1	Evaluation.....	135
5.1.2	Generalizing User Assignments .....	136
5.1.3	Embedded CLR Structures.....	136
5.1.4	Are Attribute Patterns Empirical Preference Rules?.....	137
<b>5.2</b>	<b>Case Analysis</b> .....	<b>137</b>
5.2.1	Assigning Cases One by One .....	137
5.2.2	Aggressive Case Analysis .....	138
5.2.3	A General Purpose Case Pattern Dictionary .....	139
<b>5.3</b>	<b>Noun Modifier Relationship Analysis</b> .....	<b>139</b>
5.3.1	Noun Modifier Bracketing .....	139
	Storing Pairs from Prepositional Phrases .....	139
	Branching Frequencies .....	139
	Insensitivity to the Bracketing Threshold.....	140
5.3.2	Other Noun Modifier Relationships .....	140
5.3.3	Methods for Choosing the Best NMRs .....	140
5.3.4	Taxonomic Information from NMRs .....	141
<b>5.4</b>	<b>A Unified Set of Semantic Relationships</b> .....	<b>141</b>
<b>5.5</b>	<b>Other Directions Altogether</b> .....	<b>142</b>
5.5.1	Fully Automatic Recognition of Semantic Relationships .....	142
5.5.2	Extending HAIKU with WordNet .....	142
5.5.3	Other Sources of Syntactic Information .....	143

## 5 Future Work

In this chapter I consider various related projects and extensions to the work described in the previous chapters. The first three sections deal with extensions to clause level relationship analysis (5.1), case analysis (5.2) and noun modifier relationship analysis (5.3). These extensions represent small, reasonable projects that fit within the existing TANKA framework. Crucially, HAIKU has been designed to allow many of these extensions as simple “plug-ins” that would not require significant alterations to the existing system. More significant departures appear in sections 5.4 and 5.5.

### 5.1 Clause Level Relationship Analysis

---

#### 5.1.1 Evaluation

Clause level relationship analysis has been so far evaluated on a relatively small scale. The requirement for complete, correct parse trees up to the level of clauses limits the

amount of data available for CLR analysis.<sup>1</sup> The problem is compounded by the fact that data for CLRA are likely to be sparse in average texts (as already seen in *clouds* and *small engines*).

In order to test the preference rules and compare their performance to the use of stored attribute patterns, larger experiments are needed. Such experiments could also help investigate a correlation between the syntactic verb phrase features and CLRs.

### 5.1.2 Generalizing User Assignments

The CLR analyzer extension that learns from generalized user assignments considers four attributes: tense/modality and polarity of both clauses involved in the CLR. A more fine-grained generalization on the tense/modality attribute is possible. Two clauses may differ on their tense, but have the same level of modality (stronger, weaker, none). CLRA could replace differing tenses with a variable while maintaining the modality value if it is the same for both clauses. For example, imagine the second clause from one sentence has tense/modality of *future\_simple/stronger\_modal* and the second clause from another sentence has *must\_present\_continuous/stronger\_modal*. CLRA could generalize the two attribute values to *X/stronger\_modal*.

### 5.1.3 Embedded CLR Structures

The CLR analyzer cannot hold competitions between embedded CLR structures because an embedded CLR structure has no syntactic verb phrase features. Recall sentence (21) from chapter 2.

(21) *The printer can print if the program issues the print command before the system shuts down.*

The clause identifier *\*statement1\** refers to the subtree for *the printer can print*; *\*statement2\** to the subtree for *the program issues the print command* and *\*statement3\** to *the system shuts down*. CLRA first assigns a CLR to the relationship between *\*statement2\** and *\*statement3\** resulting in CLR structure *S*. CLRA then assigns a CLR to the relationship between *\*statement1\** and *S*. The tense/modality and polarity for *\*statement1\** are *can\_present/weak\_modal* and *pos*, but the structure *S* has no distinct tense/modality or polarity values.

Similarly, the stored attribute pattern representation is not defined for embedded structures. One possibility would be to treat the attributes of the embedded structure as variables. Usually, however, variables in attribute patterns are the result of two patterns having different values for a given attribute. The variables in patterns for embedded structures would not be the result of known mismatches in attribute values and therefore

---

<sup>1</sup> I should stress that the dearth of correct parses is a weakness of the parsing technology, not of the CLRA mechanisms, although CLRA's dependence on good parsing may rightly be considered a weakness of the semantics-from-syntax philosophy.

may be overgeneralizations. To mitigate the effect of these variables, attribute patterns from embedded CLR structures could be given a lower weighting for pattern lookups. Because the CLRA implementation does not currently weight patterns, a general weighting scheme would also need to be invented and justified.

#### 5.1.4 Are Attribute Patterns Empirical Preference Rules?

HAIKU uses either stored attribute patterns or preference rules to find a CLR to suggest to the user. In section 2.5.5 I said that CLR competitions giving correct analyses also result in stored patterns, so the knowledge encoded in the preference rules is not lost when using stored patterns. That is, stored attribute patterns identify what values of the verb phrase features result in the assignment of a particular CLR—knowledge that comes from the successful application of preference rules. As more sentences are analyzed, an attribute not relevant to the assignment of a CLR is more likely to be generalized to a variable than attributes that *are* relevant. For example, if the polarity of one of the clauses is not relevant in the assignment of a particular CLR (*i.e.*, may be positive or negative), the system is more likely to assign that CLR to sentences with either polarity value than if the polarity *were* relevant.

An interesting experiment would be to test this equivalence between preference rules and attribute patterns by using stored patterns in CLR competitions. In a competition, HAIKU would look up all of the stored patterns containing the two competing CLRs (CLR<sub>1</sub> and CLR<sub>2</sub>). It would then generalize all the stored patterns containing CLR<sub>1</sub>, generalize the patterns containing CLR<sub>2</sub>, and compare the features of the input sentence to both generalizations.<sup>2</sup> The better match would win.

## 5.2 Case Analysis

---

### 5.2.1 Assigning Cases One by One

When comparing the case marker pattern of a clause to that of a previously analyzed clause, HAIKU will suggest the previous case pattern only if both CMPs have the same number of markers. If the two CMPs have different lengths, HAIKU will not suggest any cases at all, even for markers that the CMPs have in common.

Instead of giving up on the CMP, HAIKU could suggest cases marker-by-marker. If a particular marker appears in both CMPs, the corresponding case from the previous CP could be suggested. For markers in the current CMP that do not appear in the previous CMP, HAIKU could look for other instances of the marker with the current verb and

---

<sup>2</sup> The generalization of modality to stronger, weaker or none as described in 5.1.2 would be less prone to overgeneralization than the more coarse generalization of tense/modality together.

suggest those previous cases. If the marker has never appeared with the current verb, HAIKU could even check for instances of the marker with other verbs.

For example, when HAIKU is analyzing (242), it assembles the CMP  $p_{subj}-p_{obj}-at-adv$  and looks for previous CMPs with four markers. Since neither (240) nor (241) have CMPs with four markers, HAIKU asks the user to supply the case pattern.

(240) [*Bernice*  $p_{subj}/EXPR$ ] *lost* [*her keys*  $p_{obj}/OBJ$ ] [*on Tuesday*  $on/TAT$ ].

(241) [*Charles*  $p_{subj}/EXPR$ ] *lost* [*at the racetrack*  $at/LAT$ ].

(242) [*Ajax*  $p_{subj}/????$ ] *lost* [*his poem*  $p_{obj}/????$ ] [*at the club*  $at/????$ ] [*yesterday*  $adv/????$ ].

Three of the four markers in the CMP for (242) have already been encountered with the verb *lose*. HAIKU could suggest Experiencer for  $p_{subj}$ , Object for  $p_{obj}$  and LocationAt for  $at$ . The user would only have to supply one case, TimeAt for  $adv$ .

Even when HAIKU does find a perfectly matching CMP, it asks the user to choose among all of the previous case patterns that have been associated with the CMP. Often these previous case patterns have cases in common. For example, at one point in the *clouds* experiment, twelve CPs had been accumulated for the CMP  $p_{subj}-p_{obj}$  (Table 36). If a new clause appears with the CMP  $p_{subj}-p_{obj}$ , HAIKU will suggest all twelve CPs to the user.

(1) AGT-DIR	(2) AGT-EFF	(3) AGT-OBJ
(4) CONT-EXPR	(5) EXPR-CAUS	(6) EXPR-MANR
(7) EXPR-MEAS	(8) EXPR-OBJ	(9) EXPR-TAT
(10) OBJ-MEAS	(11) RECP-MEAS	(12) RECP-OBJ

---

Table 36: Twelve case patterns for  $p_{subj}-p_{obj}$

The user first looks at CP (1) considering Agent as a potential case for  $p_{subj}$ . If he decides that Agent is indeed the correct case, he must choose a case for  $p_{obj}$ . But now there are only really three CPs to choose from, since none of the others assign Agent to the subject of the sentence. Instead of suggesting twelve CPs for  $p_{subj}-p_{obj}$ , HAIKU could first present only the (five) cases corresponding to  $p_{subj}$ . If the user chooses Agent (for example), then HAIKU could suggest only the three object-marked cases Direction, Effect and Object. The disadvantage to assigning cases one by one is that slightly more user input would be required (for example, typing a number selecting each case instead of one number for a whole CP).

### 5.2.2 Aggressive Case Analysis

When HAIKU encounters a clause with a CMP that has already appeared with other clauses, it suggests *all* of the previous case patterns. Although experiments have shown that the number of accumulated CPs for any CMP grows slowly, it might be possible to

reduce user burden by making HAIKU choose a single CP (or at least a proper subset of the accumulated CPs) to suggest to the user.

The choice could be based on the number of times each CP has been previously assigned, or on some score calculated from the number of times each individual case within the CPs has occurred. Experiments would be needed to see how the case analyzer's success rate is affected, and to determine if the benefit to the user of choosing from fewer suggestions is significant.

### 5.2.3 A General Purpose Case Pattern Dictionary

The output of case analysis is a list of the verbs in a text and the cases assigned to their arguments. Many systems that use cases for semantic analysis of text require knowledge of the kinds of case roles allowed for verbs. Briscoe & Carroll (1997) identify the need for a dictionary that contains the number and categories of a predicate's arguments. HAIKU could be used as a tool to create such a dictionary. Delisle & Szpakowicz (1997) explore the construction of a dictionary of predicate-argument structures.

By combining the marker→case associations with subcategorization information, HAIKU's output could also be used to identify which cases are core for verbs (see section 3.2.4).

## 5.3 Noun Modifier Relationship Analysis

---

### 5.3.1 Noun Modifier Bracketing

#### *Storing Pairs from Prepositional Phrases*

Whenever the system brackets a list of premodifiers, it stores the reduced subbracketings to help bracket subsequent noun phrases. The system could be extended to store pairs resulting from postmodifying prepositional phrases as well. For example, for the postmodifying prepositional phrase in (243) the system would store pair (244).

(243) *a pot for soup*

(244) *(soup pot)*

Storing these extra pairs would increase the likelihood of the system finding evidence when bracketing new noun phrases.

#### *Branching Frequencies*

In section 4.7.1 I noted that left-branching triples do not always outnumber right-branching triples in a text, as has been observed by others. It will be important to continue to monitor these frequencies in future experiments to determine whether the *sparc* text is

unique in its predominance of right-branching triples. If left-branching predominance is confirmed, the system could be modified to guess left in the absence of other evidence.

### ***Insensitivity to the Bracketing Threshold***

Results presented in 4.7.1 also revealed the insensitivity of the bracketer to the value of the threshold applied to previous bracketing evidence. It is possible that a single technical text is unlikely to have both right-branching and left-branching evidence for a given triple, rendering the comparison threshold irrelevant. Future experiments may confirm this result or, if conflicting evidence is common, find a suitable default threshold value.

### **5.3.2 Other Noun Modifier Relationships**

Experiments with the noun modifier relationship analyzer have not yet uncovered the need for new NMRs. Nonetheless, it is possible to think up new ones. For example, Container and Content are currently stretched to cover part-whole relationships, which might deserve their own NMRs. The possibility of a Resemblance NMR was investigated in section 4.7.3.

Another possible addition to the NMRs would be inverse relationships. An NMR and its inverse relationship express the same semantic relationship, but the roles of modifier and head are reversed. Some of the NMRs already have inverses: Container and Content; Located and Location. It seems that some other roles might have natural inverses. Source, for example, has the paraphrase “*modifier* is the source of *compound*”. An inverse Source would have the paraphrase “*compound* is the source of *modifier*”, and could be introduced to account for (245).<sup>3</sup>

(245) *water faucet*  
*water\_faucet* is the source of *water*

Not all NMRs, however, have compelling examples for inverse relationships.

The addition of new NMRs will be left to future work. More experiments would look for noun phrases that are not covered by the existing NMRs, and provide data about the distribution of NMRs in texts. As with the cases, overrepresentation in texts might be an indication that an NMR is too general and that more specific NMRs are needed.

### **5.3.3 Methods for Choosing the Best NMRs**

Sections 4.6.4 and 4.7.2 described various methods for choosing the best NMRs from two lists of candidates. Experimental evaluation of the different methods was inconclusive. Future experiments should continue to evaluate all of the implemented methods to determine if one method is significantly better than the others. Alternatively, a more theoretically sound calculation for choosing NMRs could be sought.

---

<sup>3</sup> Without an inverse Source NMR, (245) would be assigned Product.

### 5.3.4 Taxonomic Information from NMRs

In section 4.7.3 I noted that not all NMRs should result in the generation of *isa* structures. At present the only such NMR is *Property*, for which *isa* generation is disabled. If new NMRs are added to HAIKU, it is possible that there would be other relationships (such as a *Resemblance* relationship) that should not result in *isa* structures either. The simplest solution would be to add such relationships to the list of *isa*-defeating relationships along with *Property*.

A more general solution would be to associate with each NMR what type of hierarchical structure should be automatically generated for that NMR. In this way, any taxonomic relationship could be associated with individual NMRs. The existing NMRs would all indicate an *isa* relationship except *Property*, which would indicate that no taxonomic relationship should be generated. Other taxonomic relationships could be indicated as well, such as the *islikea* relationship from section 4.7.3 or even a meronymic relationship for NMRs like *Instrument* or part-whole NMRs.

## 5.4 A Unified Set of Semantic Relationships

---

The choice of a surface representation of a sentence is often arbitrary, as illustrated by (246) and (247), both of which describe the same events.

(246) *A man was murdered yesterday with a handgun because a jealous husband returned early.*

(247) *the murder of a man yesterday with a handgun because of the early return of a jealous husband*

In both examples, a handgun is the *Instrument* of a murdering event, a jealous husband is the *Agent* of a returning event, etc. There is obviously some overlap in HAIKU's three types of semantic relationships.

Deverbal nouns such as *murder* and *return* are known to mark case-like relationships. Levi (1978) calls such nouns nominalizations, and accounts for their semantics with four case-like roles: *Act*, *Agent*, *Patient*, *Product*. HAIKU has NMRs that resemble cases to account for compounds with deverbal noun heads.

Just as some nouns participate in case-like relationships with their modifiers, stative verbs are more appropriately analyzed by assigning NMRs between their arguments. Frawley (1992) argues that there is no thematic role in (248) between *ball* and *is*, or between *red* and *is*. Rather, there is a relationship between *ball* and *red* directly, while the verb *is* is irrelevant.

(248) *The ball is red.*

By unifying HAIKU's three sets of semantic relationships into one set, it might be possible to mix and match evidence from the different levels. Lauer (1995a: 154) writes: "to interpret *garbage collection*, knowledge of the semantics of, and case roles for, the verb *collect* are needed." HAIKU may indeed have knowledge of the case roles for *collect* accumulated during case analysis. It is possible to uncover the verb root of a deverbal noun (Hull & Gomez 1996; Barker *et al.* 1997). The NMR analyzer could check stored case patterns for that verb as evidence for semantic relationship assignment.

Unifying HAIKU's semantic relationships would be a large undertaking and there is little precedent for such an all-encompassing set. In particular, validation would be a considerable endeavour.

## 5.5 Other Directions Altogether

---

The extensions described in sections 5.1, 5.2 and 5.3 could all be implemented within the existing HAIKU framework. In this section I investigate projects that represent a significant departure from TANKA principles.

### 5.5.1 Fully Automatic Recognition of Semantic Relationships

Semi-automatic analysis in HAIKU proceeds linearly through a text to avoid disorienting the user. Sentences appearing later in a text, however, might have constructions that could be used to disambiguate constructions appearing earlier. A fully automatic analyzer could process sentences and phrases in any order to arrive at a cumulative analysis of a complete text. The system could proceed in several passes (iteratively), making all of the unambiguous decisions and interpretations first and storing them. In subsequent passes more constructions could be unambiguously analyzed using structures from previous passes. The number of passes could be limited by requiring that some minimum number of relationships be recovered on each pass for processing to continue.

Several issues would require investigation in building a fully-automatic HAIKU. Would it be justifiable, for example, to use stored structures from texts already analyzed to help start the processing of a new text? Could existing knowledge resources such as WordNet replace HAIKU's user as oracle? What would be the effect of incorrect parse trees assumed to be correct? What would be the effect of incorrect stored semantic structures on future analyses?

### 5.5.2 Extending HAIKU with WordNet

HAIKU avoids semantic information associated with nouns, verbs and adjectives since such open-ended information would require a large knowledge engineering effort. Some

such knowledge is available, however, in WordNet. HAIKU could make use of WordNet in several places.<sup>4</sup>

In clause level relationship analysis the WordNet synonym sets or hypernym sets for the verbs of each of the clauses could be added to the stored attribute patterns. There may be a correlation between certain kinds of verbs (such as *cause* and *prevent*) and CLRs. Patterns with a non-nil intersection of synonym (or hypernym) sets in the verb attribute would be considered matching. It might also be possible to discover modal expressions in clauses using WordNet. Synonym sets for adverbs in a clause could be expanded until one of a small number of known modal adverbs is found (such as *certainly*, *probably*, *possibly*).

In case analysis, case marker pattern matching could be extended to match on semantics of the case fillers. For example, if two syntactic subjects from two different sentences share a hypernym, they may mark the same case.

In sections 4.4.1 and 4.7.3 I proposed using WordNet pertainyms to improve awkward noun modifier relationship paraphrases. HAIKU's NMR analyzer could also generalize NMR arguments using WordNet. Instead of looking for exact matches on modifiers and heads, HAIKU could look for intersection of their synonym sets or hypernym sets. For example, if the modifier of the current triple has no match in the stored triples, the system could look for matches using elements of the modifier's synonym set. If no previous triples match elements of the synonym set, the system could look for matches using elements of the modifier's hypernym set.

### 5.5.3 Other Sources of Syntactic Information

The experiments described in this dissertation underlined HAIKU's reliance on good parse trees from DIPETT. It would be an interesting project to adjust the system to work with other sources of syntactic information. In its current form HAIKU depends on the actual structures produced by DIPETT, but it could be modified to accept structures from other parsers, as long as they provide comparable information. For example, any parser that identifies syntactic subjects, objects and prepositional phrases attached to a verb could feed case analysis in HAIKU. Any system that identifies noun premodifiers (such as the parsers described by Voutilainen & Padró 1997 or Zhai 1997) could feed NMR analysis.

---

<sup>4</sup> Delisle (1996) describes a system that allows information from WordNet to be added to DIPETT's dictionary.