

4	Noun Modifier Relationship Analysis	85
4.1	Introduction	85
4.1.1	Noun Compounds	86
4.1.2	Semantic Relations in Noun Phrases	87
4.1.3	Recognizing Semantic Relations	89
4.2	Input Structures	92
4.2.1	Attributes and Head Noun Premodifiers	92
4.2.2	Noun Phrase Postmodifiers	93
	Relative Clauses	93
	Appositives	93
	Comparison	94
	Prepositional Phrases	94
4.3	Noun Modifier Bracketing	94
4.3.1	Subphrases and Reduced Subbracketings	95
4.3.2	Bracketing Models	96
4.3.3	A Bracketing Algorithm	96
4.3.4	Confidence in Branching Decisions	98
4.3.5	User Interaction	98
4.3.6	When Is “Left-Branching” “Not-Right-Branching”?	99
4.3.7	A Walk through Bracketing	100
4.3.8	When All Else Fails	102
	Redoing the Bracketing Interaction	102
	Ignoring Bracketing History	102
	Bracketing by Hand	103

4.4	The Noun Modifier Relationships.....	105
4.4.1	NMR Glossary.....	106
4.5	The NMR Marker Dictionary	106
4.6	Assigning NMRs	108
4.6.1	Reduced Modifiers and Heads	108
4.6.2	Modifier-Head-Marker Triples.....	109
4.6.3	Distance between Triples	110
4.6.4	The Best NMRs	111
	Absolute Frequency.....	112
	Relative Frequency.....	112
	Weighted Relative Frequency	112
4.6.5	User Interaction	113
4.6.6	Classifying Function of Premodifiers.....	115
4.7	Evaluation	116
4.7.1	Bracketer Evaluation	116
	System Performance.....	116
	The Effect of the Threshold.....	117
	Branching Frequencies	119
4.7.2	NMRA Evaluation.....	119
	System Performance.....	119
	Coverage.....	122
	User Burden.....	122
4.7.3	Some Difficulties.....	122
	Questionably Endocentric Compounds	122
	Postpositive Adjectives	124
	Adjectives in Paraphrases.....	126
4.8	An Example.....	126
4.9	Chapter Summary	131

4 Noun Modifier Relationship Analysis

Noun modifier relationships are semantic relationships between a noun and its modifiers. Syntactic analysis finds noun phrases in a sentence and provides a flat list of premodifiers and postmodifying prepositional phrases and appositives. The noun modifier relationship analyzer first brackets the flat list of premodifiers into modifier-head pairs. Next, it assigns labels to each pair. Labels are also assigned to the relationships with each postmodifying phrase.

This chapter is about HAIKU's third lobe: noun modifier relationship analysis. Since there is little syntactic evidence to guide recognition, the noun modifier relationship analyzer must pay particular attention to user interaction and learning from previous analyses.

4.1 Introduction

This introduction provides background on noun compounds, semantic relations within noun phrases and other systems that recognize these relations in texts. Section 4.2 identifies the parts of a DIPETT parse tree that are relevant to noun modifier relationship analysis (NMRA). In section 4.3 I present HAIKU's semi-automatic premodifier bracketer.

I discuss the bracketing problem and kinds of solutions. I also explain why bracketing general noun phrases is not simply an extension of noun-noun-noun bracketing. I then follow with the semi-automatic algorithm, illustrations of the various bracketer features and some difficult examples. The NMR list in section 4.4 is presented through paraphrases and examples. Section 4.5 contains a brief description of the dictionary of mappings from prepositions to NMRs. In section 4.6 I give details of the NMR assignment process: how the similarity of phrases is measured; how the system copes with conflicting previous evidence; the role of the user and user options. Generating taxonomic relationships from NMR structures is also discussed in section 4.6. Results of evaluating both the bracketer and NMR analyzer appear in section 4.7 along with some examples that are not covered by NMRA. The chapter ends with a longer example of bracketing and NMRA applied to several phrases. The example illustrates HAIKU's ability to learn from previous analyses.

4.1.1 Noun Compounds

A head noun along with a premodifying noun is often called a noun compound. Syntactically a noun compound acts as a noun: a modifier or a head may again be a compound. The NMR analyzer deals with the semantics of a particular kind of compound, namely compounds that are *transparent* and *endocentric* (Bauer 1983).

The meaning of a *transparent* compound can be derived from the meaning of its elements. For example, (152) is transparent (a *printer* that uses a *laser*), whereas (153) is *opaque* since there is no obvious direct relationship to *guinea* or to *pig*.

(152) *laser printer*

(153) *guinea pig*

An *endocentric* compound is a hyponym of its head. For example, (154) is endocentric because it is a kind of *computer*. (155) is *exocentric* because it does not refer to a kind of *brain*, but rather to a kind of person (whose brain resembles that of a bird).

(154) *desktop computer*

(155) *bird brain*

The distinction between endocentric and exocentric is not always obvious, as illustrated by (156), which does not really refer to a kind of *truck*, though it may inherit many of *truck*'s defining features. I will return to examples such as (156) in section 4.7.3.

(156) *toy truck*

Since the NMR analyzer is intended for semantic analysis of technical text, the restriction to transparent endocentric compounds should not limit the utility of the system. Experiments with the NMR analyzer (such as those described in section 4.7.2) have

found no opaque or exocentric compounds in the test texts. For these texts at least, little is lost by not dealing with opaque or exocentric compounds.

4.1.2 Semantic Relations in Noun Phrases

Most of the research on relationships between nouns and modifiers deals with noun compounds, but these relationships also hold between nouns and adjective premodifiers or postmodifying prepositional phrases. Researchers have proposed lists of semantic labels, based on the theory that a compound expresses one of a small number of covert semantic relations between its two elements.

Downing (1977) has noted that the semantic classes of the elements in a compound imply particular semantic interpretations. For example, for head nouns of class *animal*, the preferred interpretation of the modifier is *appearance* or *habitat*. If the head is *synthetic object*, the interpretation of the modifier is *purpose*. Although HAIKU does not have semantic class information for nouns and adjectives, the observation suggests that the relationships likely for a given head noun sense in a compound are restricted and tend to reappear for that sense.

Although it is common to restrict research to noun-noun compounds, Levi (1978) argues that semantics and word formation cause these compounds to constitute a heterogeneous class. Levi removes opaque compounds (such as idioms and compounds that have become lexicalized) and exocentric compounds from the list. She adds *nominal non-predicating adjectives* to the group as possible modifiers because semantically they function identically to premodifying nouns, as in (157). Levi calls the resulting set of compounds *complex nominals* to distinguish them from noun-noun compounds. She claims that unlike noun-noun compounds, complex nominals form a well defined set.

(157) [*automobile* noun] *industry*
 [*automotive* nominal non-predicating adjective] *industry*

George (1987) questions the claim that Levi's nominal adjectives can never appear in predicative position, offering counterexamples and syntactic tests that show that this *can* happen. Levi admitted the possibility, but said that constructions with non-predicating adjectives in predicative position were the result of more complex transformations, such as ellipsis or head noun deletion. George claims that not all predicative occurrences of non-predicating adjectives can be accounted for by Levi's transformations.

For example, Levi would consider the adjective *civil* in (158) non-predicating, supported by the unacceptable paraphrase in (159). Although the occurrence of *civil* in predicative position in (160) is acceptable, she would claim that it is due to a late deletion in (161) of the second occurrence of *engineer* as head noun.

(158) *civil engineers*

(159) * *the engineers are civil*

(160) *Does your firm employ both civil engineers and electrical engineers?*
No, all of our engineers are civil.

(161) *All of our engineers are civil engineers.*

One of the criticisms of attempts to interpret the semantics of compounds is that the number of possible relationships between the two components is intractable. According to Levi, however, a complex nominal is the result of deleting one of a small number of possible underlying predicates. The interpretation of a complex nominal, then, involves recovering the deleted predicate. The process is multiply ambiguous, but only over the small number of general deletable predicates. Levi lists these *recoverably deletable predicates* in lexical form, though a typical semantic form is also given. The recoverably deletable predicates appear in Table 20.

Cause	For	Use
Make	About	In
Be	Have	From

Table 20: Recoverably deletable predicates from Levi (1978)

A second kind of complex nominal is the result of predicate nominalization rather than predicate deletion. In a complex nominal formed by predicate deletion, the modifier and head correspond to arguments of the original predicate. In a complex nominal formed by nominalization, the head corresponds to the predicate itself and the modifier corresponds to either the subject or the object of the original predicate. Since the predicate persists as the head noun, complex nominals formed by predicate nominalizations are not restricted to a small number of general predicates like nominals formed by predicate deletion.

Warren (1978) attempts to identify a short list of semantic relations between the constituents of a noun-noun compound. Her *generalized verbs* are meant to represent a predicate that was deleted in the formation of the compound, much like Levi's deleted predicates. For example, a compound expressing a Resemblance relation (such as *pot hole*) is the result of deletion of the *resembles* predicate (in *hole resembles a pot*). Warren also offers a list of prepositional paraphrases for the relationship between elements in the compound. For example, a compound expressing an Origin-Object relation (such as *country boy*) is usually only paraphrased by *from* (as in *boy from the country*).

Warren's semantic relations are not a flat list, but a hierarchy. Each level of the hierarchy gives a different level of detail. The Semantic Class level (below the more general Major Semantic Class and above the more specific Main Group and Subgroup levels) is reproduced as Table 21.

Warren (1984) extends her earlier work to include adjectives as well as modifying nouns. In particular, she extracts from a corpus adjectives with explicit adjectival suffixes such as *-al*, *-an*, *-ar*, *-en*, *-ern*, *-ic*, etc., and identifies which adjectival suffixes are likely to mark which semantic classes. For example, Place adjectives end only in *-al*, *-an*, *-ar*,

Source-Result (constitute)	Copula (be)
Resemblance (be like)	Whole-Part (belong to)
Part-Whole (have in/have on)	Size-Whole (be long, wide, etc./cost/weigh)
Goal-OBJ (lead to)	Place-OBJ (be positioned in/at/on)
Time-OBJ (occur/appear at)	Origin-OBJ (be from)
Causer-Result	Motive Power-Result
Purpose (be for)	Activity-Actor (be concerned with)

Table 21: Noun-noun semantic classes from Warren (1978)

-ern, *-ese*, *-ic*, *-ish* and *-ly*. Unfortunately, the list of adjectival suffixes under consideration is small and many semantic classes are marked by a majority of them.

The semantic relations Warren presents for adjectives appear in Table 22. The similarity to the list for noun-noun compounds suggests that many adjectives and premodifying nouns can be handled by the same set of semantic relations.

Source-Result	Origin-OBJ
Result-Source	Time-OBJ
Norm-Adherent	OBJ-Time
Comparant-Compared	Affected Object-Actor
Whole-Part	Causer-Result
Part-Whole	Result-Causer
Place-OBJ	Purpose

Table 22: Adjective-noun semantic classes from Warren (1984)

4.1.3 Recognizing Semantic Relations

Programs that uncover the relationships in modifier-noun compounds often base their analysis on the semantics of the individual words, which assumes the existence of some dictionary of semantic information.

Leonard's system (1984) assigns semantic labels automatically to noun-noun compounds. The semantic relationships appear in Table 23. To arrive at an interpretation, the system proceeds through nine ordered heuristics that test semantic features of the nouns in the compound. The program depends on the semantic features encoded manually in Leonard's dictionary. These features include taxonomic information, syntactic behaviour, information about relationships between nouns and verbs, relative size of a concept in a meronymic hierarchy, etc. For example, if the modifying noun has a semantic feature *material* and the head has a semantic feature *can be composed of material*, then the semantic relationship between the two is likely Material. Experiments reveal a 76% accuracy in assigning relationships to previously unseen compounds consisting of known words (*i.e.*, words in Leonard's dictionary).

Sentence Equative	Locative Sentence Material	Locative Additive	Annex Reduplicative
----------------------	-------------------------------	----------------------	------------------------

Table 23: Noun-noun semantic relationships from Leonard (1984)

Finin's UNCLE system (1986) deals with the semantic interpretation of nominal compounds, which include both noun-noun pairs and nominal adjective-noun pairs. UNCLE produces multiple possible interpretations of a compound and assigns an appropriateness score to each. The interpretations are based on precoded semantic class information and domain-dependent frames describing the roles that can be associated with certain nouns. For example, for the compound *Chicago flight*, UNCLE gives three possible interpretations: *Chicago* fills either a Source role, a Destination role or a Stopover role. These interpretations are based on knowledge of both *Chicago* and *flight*: *Chicago* is known to be a city, and *flight* is a frame with Source, Destination and Stopover slots whose fillers are restricted to cities.

Ter Stal (1996) describes the interpretation of nominal compounds in the Plinius project. Interpretation unifies compounds with concept structures extracted from a hand-coded lexicon. For each word the lexicon contains part of speech information, "other syntactic features" (including subcategorization information, agreement features, etc.), an Underspecified Logical Form for the word, and the concept to which the word refers within an ontology. The restricted domain and corpus size (400 abstracts on mechanical properties of ceramic materials) limit the amount of hand-coding necessary, but building the ontology and lexicon still requires considerable labour and expertise. The availability of the hand-coded information allows ter Stal to skirt several linguistic problems. For example, he investigates various methods for bracketing three-word compounds but ultimately treats multi-word compounds as flat lists, since the information available in the lexicon and ontology allow direct identification of complex concepts.

In an attempt to avoid the hand-coding required by other systems, Vanderwende (1993) automatically extracts semantic features of nouns from online dictionaries. SENS (the system for Evaluating Noun Sequences) chooses the most likely semantic relation between two nouns in sequence by considering the semantic properties of each noun. The features extracted from dictionary definitions and example sentences appear in Table 24.

Abstract	Has-Subject	Material
Caused-By	Human	Means
Event	Is-For	Object-Of
Food	Location-Noun	Subject-Of
Group-Noun	Location-Of	Time
Has-Object	Made-Of	

Table 24: Semantic features extracted from dictionaries in Vanderwende (1993)

Combinations of the semantic features imply particular semantic interpretations. For example, if the head noun has the *IS-FOR* feature and the modifier matches one of the values of that feature, then the semantic relationship between head and modifier is almost certainly a What for? relationship (Purpose). SENS also attempts matching on hypernyms of the nouns (if available from the dictionary extraction) if matching on the nouns themselves fails. The semantic relations used in SENS appear in Table 25.

Who/what? (Subject of a deverbal head)	How? (Instrument)
Whom/what? (Object of a deverbal head)	What for? (Purpose)
Where? (Locative)	What kind of? (Equi/General)
When? (Time)	Made of what? (Material)
Whose? (Possessive)	What does it cause? (Causes)
	What causes it? (Caused-by)

Table 25: Semantic relations between nouns in Vanderwende (1993)

Instead of proposing his own arbitrary list, Lauer (1995a) does away with abstract semantic relations altogether. Explicit paraphrases, clauses or prepositional phrases, usually help illustrate relationships in noun-noun compounds. Lauer wants these paraphrases to be the semantic relations. He does not claim that paraphrasing as semantic analysis is easier or more difficult than assigning abstract relationship labels, and admits that the two problems are analogous. He does say that paraphrases are more concrete. Lauer's eight "concrete semantic relations" are *of*, *for*, *in*, *at*, *on*, *from*, *with*, *about*, borrowed largely from Warren's identification (1978) of prepositional paraphrases for her semantic relations. Semantic analysis assigns probabilities to each of the different possible paraphrases of a compound. These probabilities are based on the probability distribution of the relations in which the *modifier* is likely to participate and those in which the *head* is likely to participate. The frequency of prepositional phrases in a large corpus is used to estimate the probabilities.

Fabre (1996) performs semantic interpretation of noun-noun compounds using semantic characteristics of the nouns taken from WordNet (Miller 1990). Compounds that contain a deverbal noun refer to either the accomplishment of the act corresponding to the verbal root of the noun, or one of its thematic roles. If the compound does not contain a deverbal noun, a covert predicate (verb) is assumed and both nouns fill a thematic role of that covert predicate. Roles are Agent, Theme, Instrument, Locative, etc. The covert predicates are either specific to a particular compound or general enough to apply to several nouns. For example, the compound *pipeline* has the specific covert predicate Transport. More general predicates are Characterize, Constitute, Contain and Measure.

4.2 Input Structures

The input to noun modifier relationship analysis in HAIKU is a noun phrase structure (called an *entity* structure in DIPETT). The structure has a number of arguments:

```
entity( intensifier,
        determinatives,
        attributes,
        head_noun(noun( head_noun,
                        noun_modifiers( premodifiers, postmodifiers )))
        number,
        noun_phrase_postmodifiers)
```

A noun phrase may also be a proper noun or a bare possessive. Proper nouns are not analyzed for internal structure. Bare possessives have no modifiers and therefore do not participate in NMRs. Details of the entity structure appear in Barker (1997).

Prior to parsing a noun phrase, DIPETT also performs morphological analysis, so that all words arrive at NMRA in a root form (plural to singular, gerunds and participles to root verb, comparative and superlative adjectives to positive form, etc.).

The various elements of the noun phrase identify *where* the semantic relationships exist—*i.e.*, which syntactic elements are involved in a noun modifier relationship. Despite the variety of noun phrase elements that the noun phrase structure can represent, few of the elements are involved in noun modifier relationships. Only attributes, head noun premodifiers and noun phrase postmodifiers contain elements that enter into NMRs. Unfortunately, the structure of these elements offers little indication of what those relationships are.

4.2.1 Attributes and Head Noun Premodifiers

DIPETT organizes modifying elements preceding the head noun into two flat lists: *attributes* and *noun premodifiers*. Originally, these two groups corresponded exclusively to adjectives and premodifying nouns respectively, based on the assumption that all attributive position adjectives precede all premodifying nouns. This assumption is valid for *attributive-only* (non-predicating) *adjectives*, as supported by the acceptable phrase (162) and the questionable phrase (163):

(162) *a [fast adj] [desktop noun] [computer head noun]*

(163) *? a [desktop noun] [fast adj] [computer head noun]*

For non-predicating adjectives, the assumption does not hold: both (164) and (165) are acceptable. To reflect this fact, the DIPETT grammar has been changed to accept adjectives among the noun premodifiers.

(164) *a* [*personal* *adj*] [*desktop* *noun*] [*computer* *head noun*]

(165) *a* [*desktop* *noun*] [*personal* *adj*] [*computer* *head noun*]

Most noun compound research makes more subtle distinctions on premodifiers. Recall that Levi (1978) allows nouns and nominal, non-predicating adjectives as premodifiers in complex nominals, and so does Finin (1986). Lauer (1995a) points out that trying to classify adjectives as predicating and non-predicating causes computational difficulties. He restricts his compounds to sequences of “nouns acting as a single noun”, as do Downing (1977) and Leonard (1984).

An attribute or premodifier recognized by DIPETT may be any string accepted as an adjective or noun. There is no distinction between predicating and non-predicating adjectives, nor is that information readily available from an outside source (such as an online dictionary or wordlist). The NMRs and NMR analyzer may have to account for any adjective or noun premodifier, including words that act as adjectives or nouns (such as participles, gerunds, etc.)

Before assigning NMRs, the system concatenates the attributes, premodifiers and head noun into a single flat list. It must then bracket this list into local modifier-head pairs using the semi-automatic bracketing techniques from section 4.3. Each modifier-head pair receives an NMR assignment.

4.2.2 Noun Phrase Postmodifiers

DIPETT recognizes four main types of noun phrase postmodifier: the *relative clause*, *appositives*, *restricted comparison* and *prepositional phrases*.

Relative Clauses

A *relative clause* postmodifying a noun phrase can be any finite clause recognized by DIPETT. Since it is a finite clause, it is case analyzed; the noun phrase is assigned a case within the relative clause. The case structure of the relative clause provides the link between the noun phrase and the postmodifying clause, making assignment of an NMR to the postmodifier unnecessary. Noun phrases within the relative clause are analyzed separately by NMRA.

Appositives

Appositives come in many different forms and serve many different roles as postmodifiers of noun phrases. Quirk *et al.* (1985: 17.66-68) identify different features and degrees of apposition. Regardless of the type of apposition, though, all appositives are normally considered coreferent. For this reason, the same NMR (Equative) will always be assigned to the relationship between a head noun and a postmodifying appositive. Noun modifier relationships within appositives will receive NMR analysis separately.

Comparison

DIPETT allows a single restricted type of comparison to appear as a noun phrase postmodifier, illustrated by example (166).

(166) *She owns [a faster machine_{np}] [than mine_{compar}].*

Since the noun modifier relationships are not intended to cover general comparative forms, there is no NMR that is appropriate for the comparative in (166). Experiments with the NMR analyzer to date have encountered no comparatives as noun phrase postmodifiers.

Prepositional Phrases

A noun phrase may have any number of postmodifying prepositional phrases. Multiple PPs may be either explicitly conjoined, as in (167), or implicitly conjoined, as in (168). Implicitly conjoined PPs are assumed to be conjunctive.

(167) *Use the paper [[in the drawer_{pp/LOC}] or [on the counter_{pp/LOC}] conj_pp].*

(168) *the printer [[in the lab_{pp/LOC}] [on the desk_{pp/LOC}] conj_pp]*

(169) *the printer [[with the broken tray_{pp/CONT}] [in the lab_{pp/LOC}] conj_pp]*

Conjunctive conjoined prepositional phrases may express the same NMR or different NMRs, as in (168), where both PPs express the Location NMR, or (169), where the *with* PP expresses Content and the *in* PP expresses Location. It appears that disjunctive conjoined prepositional phrases are unlikely to express different NMRs, and this guess is supported by the questionable (170). Nevertheless, there is no guarantee that such examples will *not* occur in texts. Accordingly, each PP in an explicit disjunction undergoes individual NMR assignment.

(170) ? *Use the paper [[in the paper tray_{pp/LOC}] or [with the fancy border_{pp/CONT}]].*

4.3 Noun Modifier Bracketing

An NMR is a binary relationship between a noun and a modifier. Often, the head of a noun phrase is preceded by a sequence of modifiers. In this sequence modifiers may modify the head directly, or they may modify other modifiers. Before assigning NMRs, HAIKU must bracket the sequence into nested modifier-head pairs. Example (172) shows the bracketing for noun phrase (171).

(171) *dynamic high impedance microphone*

(172) *(dynamic ((high impedance) microphone))*

The bracketing problem has been investigated by Liberman & Sproat (1992), Pustejovsky *et al.* (1993), Resnik (1993) and Lauer (1995b) among others. Systems that bracket premodifiers usually deal with the problem of bracketing three nouns: two premodifiers and a head (X-Y-Z). Bracketers typically compare occurrences of X-Y with occurrences of either Y-Z or X-Z (depending on the model—see section 4.3.2). Since these pairs may occur infrequently in a text, it would be useful to generalize (X-Y-Z) and look for occurrences of the generalization. For example, Lauer (1995b) generalizes the nouns X, Y and Z to the Roget's Thesaurus categories that contain them: R_X , R_Y and R_Z . Instead of looking for other occurrences of X-Y and X-Z in the text, Lauer's bracketer looks for occurrences of U-V and U-W such that $R_U = R_X$, $R_V = R_Y$ and $R_W = R_Z$. The technique limits generalization to nouns that occur in Roget.

4.3.1 Subphrases and Reduced Subbracketings

The problem of generalization is compounded in HAIKU since it deals with sequences of more than three words and allows adjectives as premodifiers. And since the fundamental tenet of HAIKU is to avoid precoding semantic information, a semantic generalization on the words is not readily available. We need to increase the number of hits when we search for previously analyzed noun phrases to help analyze a new noun phrase.

Consider phrase (173) and a reasonable bracketing for it in (174).

(173) *dynamic high impedance vocal microphone*

(174) *(dynamic ((high impedance) (vocal microphone)))*

Each non-atomic element of each bracketed pair can be considered a subphrase of the original phrase. Given the bracketing in (174) the subphrases for phrase (173) are phrase (173) itself as well as the subphrases in (175). If the compounds are endocentric, the subphrases will be well-formed.

(175) *high impedance vocal microphone*
high impedance
vocal microphone

Each subphrase consists of one or more modifiers and a head local to the subphrase. Local heads in (173) are *microphone* (the head of three subphrases) and *impedance*. The subphrases can be generalized by reducing each modifier and modificand to their local heads. The result is a set of reduced subbracketings of the original phrase. The reduced subbracketings of bracketed phrase (174) appear in (176).

(176) *(dynamic microphone)*
(impedance microphone)
(high impedance)
(vocal microphone)

The reduced subbracketings together are a structural generalization of the original noun phrase. Instead of simply memorizing complete noun phrases and their analyses, the system stores the subbracketings. This allows it to analyze different noun phrases that have only subbracketings in common with previous noun phrases.

4.3.2 Bracketing Models

A decision whether a given sequence X-Y-Z is left-branching, as in (177), or right-branching, as in (178), may be reached in two ways.

(177) ((*laser printer*) *manual*)

(178) (*desktop (laser printer)*)

Liberman & Sproat (1992), Pustejovsky *et al.* (1993) and Resnik (1993) all compare the number of isolated occurrences of X-Y in a corpus with the number of occurrences of Y-Z. Lauer (1995b) calls this the *adjacency* model and offers a different model, the *dependency* model, which compares the number of occurrences of X-Y to the number of occurrences of X-Z (rather than Y-Z). In Lauer's bracketer, the dependency model outperforms the adjacency model. Experiments by ter Stal (1996) using both models tend to confirm Lauer's results, yet ter Stal also notes that the results are not appreciably better than always guessing left-branching.

HAIKU's bracketer uses the dependency model, comparing frequencies of X-Y to X-Z when it brackets the sequence X-Y-Z, ignoring previous occurrences of Y-Z. Consider phrase (179). Both left- and right-bracketings are possible.

(179) *small business loan*

(180) (*small (business loan)*)

(181) ((*small business*) *loan*)

Previous occurrences of *small loan* would be evidence for right-bracketing. Occurrences *small business* would be evidence for left-bracketing. Occurrences of *business loan*, on the other hand, could be evidence for *either* bracketing: (180) restricts the *loan*; (181) restricts the *business*. But both contain (*business loan*) as a reduced subbracketing—both refer to some kind of *business loan*. Therefore, occurrences of Y-Z do not count as evidence in favour of either bracketing.

4.3.3 A Bracketing Algorithm

Bracketing sequences of three elements requires a single branching decision. That branching decision leads directly to a bracketing of the compound. Bracketing sequences of more than three elements requires multiple branching decisions. Each decision does not map directly to a bracketing. Rather, decisions are interdependent.

Example (182) requires a single branching decision to arrive at the left-bracketing shown. Examples (183) and (184) both add *top* as a single modifier in front of the sequence in (182). In both the new examples *ring pin location* still branches left. But the ultimate bracketing of (183) and (184) does not directly contain the bracketing from (182). So even once *ring pin location* is known to be left-branching, it cannot be bracketed until a subsequent branching decision involving *top* has been made. A general bracketing algorithm therefore can not be a trivial iterative (or recursive) application of an algorithm for bracketing sequences of three elements.

(182) ((*ring pin*) *location*)

(183) (((*top ring*) *pin*) *location*)

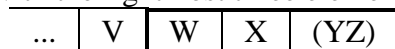
(184) ((*top (ring pin)*) *location*)

HAIKU's algorithm for noun premodifier bracketing handles modifier sequences of any length by dealing with a sliding/expanding window of three elements at a time.¹ Depending on the results of individual branching decisions, the system may have to delay bracketing locally within the window until later branching decisions have been made.

- 1 Start with the rightmost three elements, X-Y-Z.



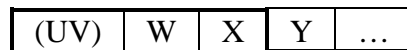
- 2a If X-Y-Z is confidently right-branching (see section 4.3.4), bracket it X-(YZ) and restart the algorithm with the rightmost three elements W-X-(YZ).



- 2b If X-Y-Z is confidently left-branching (see sections 4.3.4 and 4.3.6), move the window one element to the left and repeat the algorithm with W-X-Y. Note that X-Y-Z being confidently left-branching does not necessarily mean that X and Y can be immediately bracketed together as (XY)-Z; the bracketing ((WX)Y)-Z also has X-Y-Z left-branching.



- 3 When the leftmost element in the whole sequence appears in the window, a left-branching sequence U-V-W can be left-bracketed (UV)-W; restart with the three-element window expanded to include the next element to the right in the sequence.



When bracketing is complete, each reduced subbracketing is stored to assist future bracketing. The entire bracketed phrase is also stored, and can be looked up if the same noun phrase occurs again.

¹ An element is a word or a bracketed pair of elements.

4.3.4 Confidence in Branching Decisions

The bracketing algorithm needs to know whether subsequences X-Y-Z in the modifier sequence are confidently left-branching or right-branching.

The sequence X-Y-Z is confidently right-branching when Y is an adjective, since adjectives are almost always the modifier in a modifier-head pair. There are exceptions, where the adjective appears in the head position of modifier-head pairs. Section 4.3.8 addresses bracketing exceptions such as (185) and (186).

(185) ((*machine readable*) dictionary)

(186) ((*ill educated*) man)

For any other sequence of three elements X-Y-Z, the bracketer reduces X, Y and Z to their local heads X_h , Y_h and Z_h . The sequence X-Y-Z is considered confidently right-branching if the frequency of previous occurrences of the reduced subbracketing ($X_h Z_h$) is greater than the frequency of previous occurrences of the reduced subbracketing ($X_h Y_h$) times a threshold value. If ($X_h Y_h$) has occurred more frequently than ($X_h Z_h$) times the threshold, X-Y-Z is confidently left-branching.

In the absence of such frequency data, the algorithm consults the user. A fully automatic solution could just assume left-branching, based on the results of Lauer & Dras (1994) and ter Stal (1996) that left-branching is roughly twice as common as right-branching. These results, however, were based on observations of noun-noun-noun sequences. For longer compounds and compounds with adjectives, the ratio of left-branching to right-branching compounds might differ (see section 4.7.1).

4.3.5 User Interaction

When the system cannot find sufficient evidence in favour of right-branching or left-branching, it asks the user to supply the decision. Such decisions may be difficult and unintuitive for users. There are several ways to lessen the burden.

- ask only yes-no questions about right-branching—don't ask the user to supply bracketing information directly.

good: in the context of *copper soup pot*,
does *copper soup* make sense?

bad: does *copper soup pot* bracket left or right?

- phrase questions in the context of three individual words by using subphrase reductions.

good: in the context of *steel soup pot*,
does *steel soup* make sense?

bad: in the context of *steel tomato soup pot cover*,
does *steel tomato soup pot* make sense?

- ask the user only about the acceptability of X-Y; do not ask the user to compare X-Y and X-Z—it is possible for both X-Y and X-Z to be unacceptable if X-Y-Z is in the middle of a modifier sequence, which would make the user’s decision much more difficult.

good: in the context of *steel tomato soup*,
does *steel tomato* make sense?

bad: in the context of *steel tomato soup*,
which makes more sense: *steel tomato* or *steel soup*?

Using these principles, a ‘yes’ answer to any question will provide confident left-branching; a ‘no’ answer will mean confident right-branching.

4.3.6 When Is “Left-Branching” “Not-Right-Branching”?

I have been using the terms *right-branching* and *left-branching* for subsequences of a premodifier sequence being bracketed. Although right-branching was appropriate in the context, left-branching was less so. Consider phrase (187).

(187) *French onion soup bowl*

The bracketing algorithm starts with the rightmost sequence of three elements *onion soup bowl*, which is *not* right-branching. But the subphrase *onion soup bowl* in the context of (187) is not immediately left-branching, since *onion soup* is the head of the right-branching compound *French onion soup*, as shown graphically in Figure 7. Nonetheless, preferring simplicity over exactness, I refer to not-right-branching subsequences as left-branching.

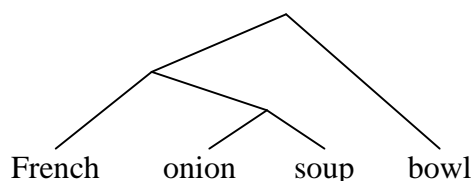


Figure 7: Branching for (187)

providing confidence in left-branching. Move the window one element to the left and restart the algorithm.

wooden	French	onion	soup	bowl	handle
--------	--------	-------	------	------	--------

Neither (*French onion*) nor (*French soup*) have occurred previously. Ask the user if *French onion* makes sense in the context of *French onion soup*. The user answers ‘no’, so the sequence is confidently right-branching. Bracket (*onion soup*) and expand the window to include the next element to the left.

wooden	French	(onion soup)	bowl	handle
--------	--------	--------------	------	--------

French is an adjective, so *wooden-French-(onion soup)* is confidently right-branching. Bracket (*French (onion soup)*). Since there are no more elements to the left of *wooden*, expand the window to include the next element to the right.

wooden	(French (onion soup))	bowl	handle
--------	-----------------------	------	--------

Neither (*wooden bowl*) nor (*wooden soup*), which is the reduction of *wooden-(French (onion soup))*, have occurred previously. (*French (onion soup)*) is obviously not an adjective, so there is no confidence in either right-branching or left-branching. Ask the user if *wooden soup* makes sense in the context of *wooden soup bowl*, which is the reduction of *wooden-(French (onion soup))-bowl*. The user answers ‘no’, providing confidence in right-branching. Bracket *wooden-(French (onion soup))-bowl* as *wooden-((French (onion soup)) bowl)*. Since there are no more elements to the left of *wooden*, expand the window to include the next element the right.

wooden	((French (onion soup)) bowl)	handle
--------	------------------------------	--------

(*wooden bowl*), which is the reduction of *wooden-((French (onion soup)) bowl)*, has not occurred previously; (*wooden handle*) has occurred previously as a reduction of (*wooden (pot handle)*). So *wooden-((French (onion soup)) bowl)-handle* is confidently right-branching. Bracket (*((French (onion soup)) bowl) handle*).

wooden	((((French (onion soup)) bowl) handle))
--------	---

Since there are only two remaining elements in the sequence, bracketing is trivial:

(wooden (((French (onion soup)) bowl) handle))
--

Finally, the system stores all of the reduced subbracketings (191) for future processing. It also stores the complete bracketing (192): if the phrase is ever encountered in its entirety, the bracketer can simply look up the complete bracketing.

(191) (*onion soup*)
 (*French soup*)
 (*soup bowl*)
 (*bowl handle*)
 (*wooden handle*)

(192) (*wooden (((French (onion soup)) bowl) handle)*)

4.3.8 When All Else Fails

The bracketer may make incorrect branching decisions based on previous analyses. There may be head-position adjectives in the premodifier sequence (see section 4.3.4). There may be bracketings in the text that conflict, or the user may simply have incorrectly answered one of the branching questions.

Redoing the Bracketing Interaction

The semi-automatic bracketer always submits the final bracketing to the user for approval. The user may accept the bracketing, or choose from several other options. The simplest option is to repeat the bracketing interaction using the same evidence as the first time through the process. This option is useful for redoing the bracketing when questions may have been answered incorrectly.

Ignoring Bracketing History

Certain previous modifier-head phrases may “fool” the bracketer. In this situation, no matter how the user answers questions, the final bracketing is incorrect. The user must turn off the bracketing history and redo the interaction as though there were no previous examples. For example, assume phrases (193) has already been bracketed.

(193) (*metric (socket wrench)*)

(194) *metric system wrench*

When bracketing (194) the system compares the number of previous occurrences of *metric system* (none) with the number of occurrences of *metric wrench* (one). It incorrectly concludes that (194) is confidently right-branching and does not ask the user to make the branching decision. The user can redo the interaction with history off to arrive at the correct bracketing, as shown in Figure 9.

Evaluation of the bracketer on complete English texts (see section 4.7.1) has shown that situations where previous analyses mislead the bracketer are rare. In the *small engines* experiment history had to be turned off only twice in 118 complete interactions.



Implementation Note

History *on/off* is a parameter that can be set in user profiles for HAIKU. If history is *on* for the current session, the user can turn it *off* for an interaction, but it is automatically turned back *on* for subsequent noun phrases. If the user has history set *off* in the profile, it can be turned *on* for individual interactions, but is automatically turned *off* for subsequent noun phrases.

(198) ((*badly worn*) *cylinder*)

(199) ((*badly worn*) (*valve seat*))

In each phrase, the intensifier (*fully*, *badly*) was misparsed as an adjective. A leading adverb should be parsed as an *intensifier* (in the entity structure—see section 4.2). Intensifiers should be concatenated with the first adjective and treated as a single modifier in NMRA (since there are no NMRs to handle the relationship between intensifier and adjective).

4.4 The Noun Modifier Relationships

Table 26 lists HAIKU's noun modifier relationships. The list was constructed after carefully considering similar lists found in literature on the semantics of noun compounds (section 4.1).

Agent (AGT)	Instrument (INST)	Property (PROP)
Beneficiary (BENF)	Located (LED)	Purpose (PURP)
Cause (CAUS)	Location (LOC)	Result (RESU)
Container (CTN)	Material (MATR)	Stative (STAT)
Content (CONT)	Object (OBJ)	Source (SRC)
Destination (DEST)	Possessor (POSS)	Time (TIME)
Equative (EQUA)	Product (PROD)	Topic (TOP)

Table 26: The noun modifier relationships (with abbreviations)

Some comments on the list are in order. First, some researchers name the semantic relationships with two role labels. For example, a PartOf relationship might be named Part-Whole, with the modifier labeled Part and the head labeled Whole. This could be justified. If a modifier-head pair hides some deleted predicate (see Levi 1978), then there may not be a single relationship between the modifier and the noun, but rather one semantic relationship between the modifier and the deleted predicate and one between the head and the deleted predicate. Nonetheless, this complex set of relationships can be captured by a single abstract label for the relationship, even if all the details are not explicit in the label. For example, a PartOf relationship label denotes a Part and implies a Whole.

Second, many lists of semantic relationships mark the direction of the relationship by including separate labels for each direction. For example, a Part-Whole label (modifier: Part, head: Whole) might have a corresponding Whole-Part label (modifier: Whole, head: Part) in some list. While this is justified for some relationships (for example, Located and Location from Table 26), not all relationships have such natural inverses. Others' lists (for example, Warren 1978; 1984) confirm the asymmetry. Rather than introduce new

labels exhaustively for the inverses, I will be conservative and add them individually if experiments reveal the need (see also section 5.3.2).

Finally, although the literature that has inspired the NMRs usually rules out certain kinds of modifiers—some ignore predicative adjectives, some ignore adjectives altogether, some avoid nominalizations, some appositional (karmadharaya) compounds—the TANKA context allows no such luxury. DIPETT passes all different kinds of modifiers to NMRA, and distinctions such as the non-predicating versus predicating feature of adjectives are not available anywhere. For these reasons, the list of NMRs contains some relationships not always found elsewhere. Property covers predicative adjectives; Equative is used for appositional compounds and Stative for other possibly copula-derived compounds. The case-like NMRs (Agent, Beneficiary, Cause, etc.) allow for nominalizations.

4.4.1 NMR Glossary

The glossary (Table 27) contains paraphrases and example compounds for the NMRs. Following the tradition in the study of noun compound semantics, the paraphrases act as definitions and can be used to check the acceptability of a semantic interpretation of a compound. The paraphrases serve as definitions in this section and as help during user interactions (as illustrated in section 4.6.5).

In the analyzer, paraphrases with adjectives may feel awkward; they could be improved by replacing adjectives with their WordNet pertainyms, giving, for example, (202) as the paraphrase of (200) instead of the infelicitous (201).

(200) *charitable donation*

(201) *charitable benefits from charitable donation*

(202) *charity benefits from charitable donation*

4.5 The NMR Marker Dictionary

Unlike clause level relationships and cases, noun modifier relationships are not always marked by a lexical or syntactic element. NMRs between heads and postmodifiers are marked lexically by a preposition or syntactically as appositives. Between premodifiers and heads, no such NMR markers exist.³ The preposition attaching a postmodifier to its head can be used to reduce the number of potential NMRs, based on the assumption that each preposition marks a subset of the NMRs.

³ I do not consider simple adjacency as a syntactic NMR marker, though that argument could be made. A *bracketing* is a marking in the sense that it indicates which elements participate in semantic relationships. The bracketing itself, however, does not help disambiguate in NMR assignment.

NMR	Paraphrase	Examples
Agent	<i>compound is performed by modifier</i>	<i>student protest, band concert, military assault</i>
Beneficiary	<i>modifier benefits from compound</i>	<i>student price, charitable donation</i>
Cause	<i>modifier causes compound</i>	<i>exam anxiety, overdue fine</i>
Container	<i>modifier contains compound</i>	<i>printer tray, flood water, film music, story idea</i>
Content	<i>modifier is contained in compound</i>	<i>paper tray, eviction notice, oil pan</i>
Destination	<i>modifier is the destination of compound</i>	<i>game bus, exit route, entrance stairs</i>
Equative	<i>modifier is also head</i>	<i>composer arranger, hinge joint, player coach</i>
Instrumental	<i>modifier is used in compound</i>	<i>electron microscope, diesel engine, laser printer</i>
Located	<i>modifier is located at compound</i>	<i>building site, home town, solar system</i>
Location	<i>modifier is the location of compound</i>	<i>lab printer, internal combustion, desert storm</i>
Material	<i>compound is made of modifier</i>	<i>carbon deposit, gingerbread man, water vapour</i>
Object	<i>modifier is acted on by compound</i>	<i>engine repair, horse doctor</i>
Possessor	<i>modifier has compound</i>	<i>national debt, student loan, company car</i>
Product	<i>modifier is a product of compound</i>	<i>automobile factory, light bulb, colour printer</i>
Property	<i>compound is modifier</i>	<i>blue car, big house, fast computer</i>
Purpose	<i>compound is meant for modifier</i>	<i>concert hall, soup pot, grinding abrasive</i>
Result	<i>modifier is a result of compound</i>	<i>storm cloud, cold virus, death penalty</i>
Source	<i>modifier is the source of compound</i>	<i>foreign capital, chest pain, north wind</i>
Stative	<i>compound is in a state of modifier</i>	<i>live wire, sleeping dog, charged battery</i>
Time	<i>modifier is the time of compound</i>	<i>winter semester, late supper, morning class</i>
Topic	<i>compound is concerned with modifier</i>	<i>computer expert, safety standard, horror novel</i>

Table 27: NMRs with paraphrases and examples

The NMR marker dictionary contains an entry for each of 46 atomic and phrasal prepositions. These prepositions were taken from several online word lists. The lists were first merged and then edited by hand to remove noise (including archaic words, uncommon foreign language prepositions, errors, etc.). For each final entry the various English dictionary senses of the preposition were mapped to NMRs by hand. This step constitutes hand-coding of semantic information, but since the list of prepositions is small, the mapping was not a difficult knowledge engineering task. I enlisted a secondary school student to build the dictionary; he completed the task in just twenty intensive hours.

The total number of mappings in the NMR marker dictionary is 126, meaning that each preposition marks on average 2.7 NMRs. 15 prepositions are mapped to a single NMR, 14 are mapped to only two. The preposition *of* is the least discriminating, mapping to 12 NMRs. A sample of the entries in the NMR marker dictionary appears in Appendix III.

The marker dictionary was constructed carefully, but it is possible that there are missing entries. Identifying new mappings in examples from texts would be simple: they consist of those prepositional phrases for which the dictionary was consulted but for which the user supplied the correct NMR.

4.6 Assigning NMRs

The process of assigning NMRs is as follows. For the modifier and head under consideration, find the most similar previously analyzed instances. Create a list (or lists) of the NMRs that were assigned to these previous instances. Suggest the best NMRs from the list(s) to the user. The user may accept the system's suggestion or supply a different NMR.

4.6.1 Reduced Modifiers and Heads

After bracketing, each non-atomic element of a bracketed pair is considered a subphrase of the original phrase, as described in section 4.3.1. Consider again the bracketed phrase (172), reproduced here with its subphrases (203), (204) and (205).

(172) *(dynamic ((high impedance) microphone))*

(203) *high impedance*

(204) *(high impedance) microphone*

(205) *dynamic (high impedance microphone)*

Each subphrase consists of a modifier (possibly compound, as in (204)) and a head (possibly compound, as in (205)). The NMR analyzer assigns an NMR to the modifier-head pair that makes up each subphrase.

Once an NMR has been assigned, the system must store the assignment to help automate future processing. Instead of memorizing complete noun phrases (or even complete subphrases), the system reduces compound modifiers and compound heads to their own local heads and stores these reduced pairs with their assigned NMR. For example, (206) and (207) have the reduced pair (*dynamic microphone*) in common. So if (206) has already been analyzed, its analysis can be used to assist in the analysis of (207).

(206) (*dynamic ((high impedance) microphone)*)

(207) (*((dynamic (cardioid microphone)) diaphragm)*)

Although the system stores *reduced* pairs for future processing, it assigns NMR labels to *complete* modifier-head sequences. Consider example (208).

(208) (*small (gasoline engine)*)

There are two NMRs for (208), since there are two modifier-head pairs:

(209) *gasoline engine*

(210) *small (gasoline engine)*

The NMR for (209) could be Instrument, since *gasoline* is used by *gasoline engine*. Note that *gasoline* is an instrument of *gasoline engine*, not *engine* in general. The NMR for (210) is Property: *small* is a property of *small gasoline engine*, but not generally a property of *engine* or even of *gasoline engine*. When an argument name is built from more than one word, the individual words are concatenated with an underscore. The result of NMR analysis for (208) would be:

gasoline is used in *gasoline_engine*
small_gasoline_engine is *small*

4.6.2 Modifier-Head-Marker Triples

Section 4.2 identified three kinds of construction that require NMR assignments: the modifier-head pairs from the bracketed premodifier sequence; postmodifying prepositional phrases; appositives.

These three kinds of input can be generalized to a single form—a triple consisting of modifier, head and marker (M, H, Mk). For premodifiers, Mk is the symbol `nil`, since no lexical item links the premodifier to the head. For postmodifying prepositional phrases Mk is the preposition. For appositives, Mk is the symbol `appos`. The (M, H, Mk) triples for examples (211), (212) and (213) appear in Table 28.

	<i>modifier</i>	<i>head</i>	<i>marker</i>
(211)	monitor	cable	nil
	monitor_cable	plug	nil
(212)	chocolate	cake	nil
	large	piece	nil
	chocolate_cake	large_piece	of
(213)	young	people	nil
	young_people	friend	to
	friend	brother	appos

Table 28: (M, H, Mk) triples for (211), (212) and (213)

(211) *monitor cable plug*

(212) *large piece of chocolate cake*

(213) *my brother, a friend to all young people*

To assign an NMR to a triple (M, H, Mk), the system attempts to find previous triples whose distance to the current triple is minimal. The NMRs assigned to previous similar triples comprise lists of candidate NMRs. The analyzer then finds what it considers the best NMR from these lists of candidates and presents it to the user for approval. Appositives are automatically assigned Equative.

4.6.3 Distance between Triples

The distance between two triples is a measure of the degree to which their modifiers, heads and markers match. Table 29 gives the eight different values for distance used in NMRA. An underscore in a previous triple means that the modifier, head or marker need not be the same as the corresponding modifier, head or marker in the current triple.

The analyzer looks for previous triples at the lower distances before attempting to find triples at higher distances. For example, it will try to find identical triples (distance 0) before trying to find triples whose markers do not match (distance 1, 2, ...).

Several things about the distance measures require explanation. First, a preposition is assumed to be more similar to a nil marker (distance 1) than to a different preposition. The nil marker could be considered a variable, which is not known to be different from the marker in an overtly marked pair. For example, (214) could be paraphrased using the marker *about* from (215) or the marker *on* from (216). (215) and (216), on the other hand, are known to have different markers.

(214) *computer encyclopedia*

(215) *encyclopedia about computers*

(216) *encyclopedia on computer*

Next, no evidence suggests that triples with matching modifiers are more similar or less similar than triples with matching heads. Consequently, previous triples with matching modifiers but different heads are at the same distance as previous triples with matching heads but different modifiers (distances 3 and 6).

Triples with matching prepositional markers (distance 4) *are* considered more similar than triples with matching modifiers or heads only. A preposition is an overt indicator of the relationship between modifier and head (see Quirk 1985: chapter 9) so a correlation is more likely between the preposition and the NMR than between a given modifier or head and the NMR.

If there are no matching triples at a distance of 4 or less and the marker is a preposition, HAIKU consults the NMR marker dictionary for candidate NMRs (represented by `nrmr(...)`) in the distance 5 row of Table 29).

4.6.4 The Best NMRs

The candidate NMRs are all NMRs previously assigned to (M, H, Mk) triples at a minimum distance from the triple under analysis. If the minimum distance is 3 or 6, there may be two candidate lists: \mathbf{L}_M contains the NMRs previously assigned to triples with matching M, \mathbf{L}_H —with matching H. HAIKU attempts to choose a set \mathbf{R} of NMRs to suggest to the user as the best candidates for the current triple.

If there is one list \mathbf{L} of candidate NMRs, \mathbf{R} contains the NMR (or NMRs) that occur most frequently in \mathbf{L} . For two lists \mathbf{L}_M and \mathbf{L}_H , \mathbf{R} could be found in several ways.

Suppose the modifier-head pair in example (217) has never been seen before, but that

<i>dist</i>	<i>current triple</i>	<i>previous triple</i>	<i>example</i>	
0	(M, H, Mk)	(M, H, Mk)	<i>wall beside a garden</i>	<i>wall beside a garden</i>
1	(M, H, <prep>)	(M, H, nil)	<i>wall beside a garden</i>	<i>garden wall</i>
2	(M, H, Mk)	(M, H, _)	<i>wall beside a garden</i>	<i>wall around a garden</i>
3	(M, H, Mk)	(M, _, Mk) or (_, H, Mk)	<i>pile of garbage</i>	<i>pile of sweaters</i>
4	(M, H, <prep>)	(_, _, <prep>)	<i>pile of garbage</i>	<i>house of bricks</i>
5	(M, H, <prep>)	(_, _, _)	<i>ice in the cup</i>	<code>nrmr(in, [ctn, ..., time])</code>
6	(M, H, Mk)	(M, _, _) or (_, H, _)	<i>wall beside a garden</i>	<i>garden fence</i>
7	(M, H, Mk)	(_, _, _)	<i>wall beside a garden</i>	<i>pile of garbage</i>

Table 29: Measures of distance between triples

front has appeared as the modifier in three other pairs (218) and *panel* has appeared as the head in nine other pairs (219).

(217) *front panel*

(218) [*front cover* *LOC*], [*front line* *LOC*], [*front plate* *LOC*]

(219) [*computer panel* *CTN*], [*control panel* *PURP*], [*control panel* *PURP*],
 [*glass panel* *MATR*], [*plastic panel* *MATR*], [*plastic panel* *MATR*], [*side panel* *LOC*],
 [*steel panel* *MATR*], [*wood panel* *MATR*]

Absolute Frequency

One possibility is to take **R** to contain the most frequent NMR(s) in $\mathbf{L}_M \cup \mathbf{L}_H$. Table 30 shows the absolute frequencies of each of the NMRs in $\mathbf{L}_M \cup \mathbf{L}_H$. Material has the highest absolute frequency in (218) and (219). Using absolute frequency, NMRA would suggest the following analysis of (217) to the user:

front_panel is made of *front*

<i>NMR</i>	<i>absolute frequency</i>
Container	1
Location	4
Material	5
Purpose	2

Table 30: Best NMR for (217) using the absolute frequency method

Relative Frequency

The absolute frequency method has a bias towards NMRs in the larger of the two lists. Alternatively, HAIKU could prefer NMRs with the highest relative frequency in their lists. If there is less variety in the NMRs in \mathbf{L}_M than in \mathbf{L}_H , M might be a more consistent indicator of NMR than H (or vice versa). Table 31 shows the relative frequency of each NMR *i* in each list calculated as $freq(i \in \mathbf{L}_M)/|\mathbf{L}_M|$ and $freq(i \in \mathbf{L}_H)/|\mathbf{L}_H|$.

<i>NMR</i>	<i>relative frequency in L_M</i>	<i>relative frequency in L_H</i>
Container	0.0	0.11
Location	1.0	0.11
Material	0.0	0.56
Purpose	0.0	0.22

Table 31: Best NMR for (217) using the relative frequency method

R contains the NMR(s) with the highest relative frequency in either \mathbf{L}_M or \mathbf{L}_H . There is a potential bias, however, for smaller lists (a single NMR in a list always has the highest

relative frequency). Using absolute frequency, NMRA would suggest the following analysis of (217) to the user:

front is the location of *front_panel*

Weighted Relative Frequency

The relative frequency method might be going too far in ignoring absolute frequency altogether. A third possibility would be to combine absolute and relative frequencies. Each NMR i is assigned a score s_i calculated as:

$$s_i = \frac{\text{freq}(i \in \mathbf{L}_M)^2}{|\mathbf{L}_M|} + \frac{\text{freq}(i \in \mathbf{L}_H)^2}{|\mathbf{L}_H|}$$

Table 32 shows the weighted relative frequency score for each of the NMRs in $\mathbf{L}_M \cup \mathbf{L}_H$.

<i>NMR</i>	<i>weighted score</i>
Container	0.11
Location	3.11
Material	2.78
Purpose	0.44

Table 32: Best NMR for (217) using weighted relative frequency scores

R would contain the NMR(s) with the highest score. This method was used in the *small engines* experiment. Experimental results appear in section 4.7.2.

4.6.5 User Interaction

For each NMR assignment the NMR analyzer asks the user's approval showing the paraphrases from section 4.4.1. The user has several options. Most often over the course of a session, the user accepts the suggestion. Alternatively, he may supply an NMR directly, ask for a list of NMR paraphrases using the current modifier and head, or even create a new NMR. Figure 11 illustrates use of the `list` feature for example (220).

(220) *steel casing*

```

String (220)      steel casing

HAIKU: Noun Modifier Relationship Analysis of current input ...

Match type 7: (_, _, _)

For the phrase 'steel casing'
There is a relationship between
  (steel) and (steel_casing).

> Please enter a valid NMR label ('a' to abort): list

Agent (agt): steel_casing is performed by steel
Beneficiary (benf): steel benefits from steel_casing
Cause (caus): steel causes steel_casing
Container (ctn): steel contains steel_casing
Content (cont): steel is contained in steel_casing
Destination (dest): steel is the destination of steel_casing
Equative (equa): steel is also casing
Instrument (inst): steel is used in steel_casing
Located (led): steel is located at steel_casing
Location (loc): steel is the location of steel_casing
Material (matr): steel_casing is made of steel
Object (obj): steel is acted on by steel_casing
Possessor (poss): steel has steel_casing
Product (prod): steel is a product of steel_casing
Property (prop): steel_casing is steel
Purpose (purp): steel_casing is meant for steel
Result (resu): steel is a result of steel_casing
Source (src): steel is the source of steel_casing
State (stat): steel_casing is in a state of steel
Time (time): steel is the time of steel_casing
Topic (top): steel_casing is concerned with steel

> Please enter a valid NMR label ('a' to abort): matr

> For the phrase 'steel casing'
Do you accept the assignment:
Material (matr): steel_casing is made of steel

[n/a/<nmr>/Y] Y

```

Figure 11: The *list* feature for example (220)

HAIKU also allows the user to add a new NMR. Figure 12 shows an interaction using the *create* feature for example (221). The *create* feature has a particularly unfriendly interface, requiring the user to enter such things as the NMR argument ordering for paraphrases. This feature is considered the last resort for particularly tricky phrases.⁴

(221) *water faucet*

⁴ See section 5.3 for a discussion of the possibility of adding NMRs such as *Inv_Source*.

```

String (221)      water faucet

HAIKU: Noun Modifier Relationship Analysis of current input ...

Match type 7: ( _, _ , _ )

For the phrase 'water faucet'
There is a relationship between
  (water) and (water_faucet).

> Please enter a valid NMR label ('a' to abort): create

You're not supposed to create new NMRs.

> Please enter the NMR Label (3-4 lower case letters): isrc
> Please enter the NMR Name: Inv Source
> Please enter the NMR argument order (mh, cm, etc.): cm
> Please enter the NMR paraphrase: is the source of

I'm about to add your NMR to the NMR list:
  t_nmr_names(isrc,'Inv_Source',cm,' is the source of ')
This is your last chance to back down.
> Do you accept the new NMR [y/N]? Y

New NMR saved (but I'm not happy about it).

> For the phrase 'water faucet'
Do you accept the assignment:
  Inv_Source (isrc): water_faucet is the source of water

  [n/a/<nmr>/Y] Y

```

Figure 12: The *create* feature for example (221)

4.6.6 Classifying Function of Premodifiers

Since NMR analysis deals with endocentric compounds, it is possible to recover a taxonomic relationship from modifier-head-marker triples with a `nil` marker (*i.e.*, those that come from compounds). The reduced pairs for example (222) appear in (223).

(222) (*colour (laser printer)*)

(223) (*laser printer*)
 (*colour printer*)

These pairs result in the additional HAIKU output:

```

laser_printer isa printer
colour_laser_printer isa laser_printer

```

The `isa` relationships are automatically generated for all of the NMRs with one somewhat arbitrary exception. Property relationships do not result in `isa` structures, because properties do not introduce hyponyms: a *small house* is not generally considered a subclass of *house*, for example. A more general solution would be to associate a feature with each NMR giving its “`isa` pattern”. This simple extension is explored in section 4.7.3 and again in section 5.3.

4.7 Evaluation

In this section I give results of two experimental evaluations of NMR analysis. The *sparc* experiment applied the NMR analyzer (including the bracketer) to the first 500 non-trivial noun phrases in the *sparc* text. In this context a non-trivial noun phrase has at least one premodifier (adjective or noun) or postmodifying prepositional phrase. Bracketing the 500 noun phrases in the test produced 637 bracketed pairs. These bracketed pairs along with 129 postmodifiers were assigned 766 NMR labels. The second experiment is the *small engines* experiment. Bracketing resulted in 733 premodifier-head pairs. Along with 153 postmodifiers, a total of 886 NMRs were assigned.

4.7.1 Bracketer Evaluation

System Performance

In both experiments, most of the modifier-head pairs occurred in noun phrases with a single premodifier and head. These simple compounds required no bracketing decisions. In the *sparc* experiment, the 637 pairs required 194 bracketing decisions. The system made 122 (63%) of these decisions correctly, with the rest made by the user. Of the 72 user decisions, 47 (65%) were required during the first half of the experiment with only 25 in the second half. The running totals of user and system bracketing decisions appear in Figure 13.



Implementation Note

The NMR analyzer stores three kinds of structures: `nmsDict` structures (containing reduced pairs that have occurred in bracketings and the number of times they have occurred); `nmsDictWhole` structures (complete flat lists of premodifier-head sequences and their corresponding bracketed forms); `nmrDict` structures (M-H-Mk triples with reduced M and H along with the NMRs assigned to them). These are the structures used to assist processing new phrases. At the end of a session the user has the option of saving the structures in a file. At the beginning of a session the user may load any number of such files of structures to seed the new session.

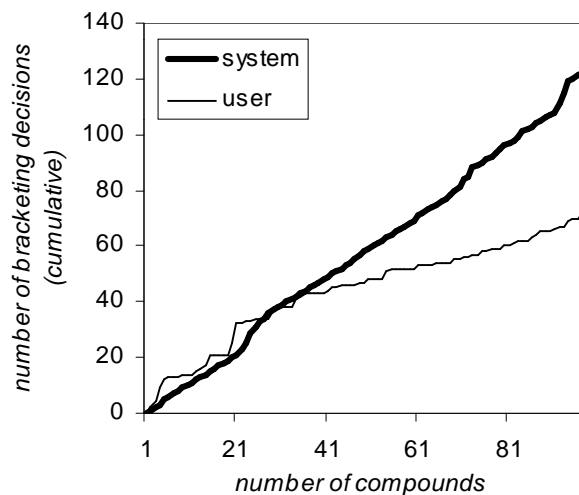


Figure 13: Bracketing decisions in the sparcs experiment

The *small engines* experiment required 164 bracketing decisions. The system made 101 (62%) decisions correctly, with the rest made by the user. Due to the consistency of the terminology in the *small engines* text the cumulative number of decisions taken automatically by the bracketer was always greater than the number required from the user. The running totals for user and system bracketing decisions appear in Figure 14.

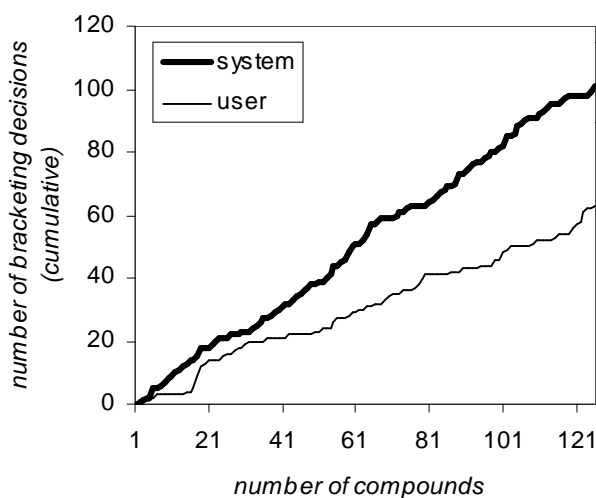


Figure 14: Bracketing decisions in the small engines experiment

The Effect of the Threshold

As explained in section 4.3.4, the bracketer determines that a given sequence X-Y-Z is confidently right-branching if the subbracketing ($X_h Z_h$) has previously occurred more

frequently than the subbracketing $(X_h Y_h)$ by a factor of N , where N is a threshold that can be set by the user. In the absence of sufficient evidence, the system asks the user questions to help acquire right-branching information.

If the value of the threshold is set high, the number of previous occurrences of $(X_h Z_h)$ must greatly outweigh the number of occurrences of $(X_h Y_h)$ for the system to assume right-branching. So high values of the threshold cause the system to be more conservative. Low values of the threshold (close to 1.0), make the system more aggressive: HAIKU requires less evidence to commit to a branching decision automatically.

The bracketer was run on the *sparc* phrases twelve times with different threshold values. As expected, the number of system decisions, both correct and incorrect, was highest for low threshold values (see Figure 15). For higher threshold values, the number of incorrect system decisions decreased, but so did the number of correct decisions, the extra decisions being given to the user.

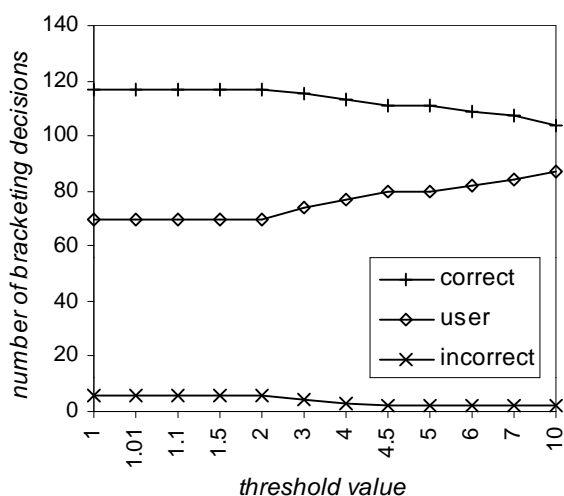


Figure 15: The effect of different threshold values on branching decisions for the *sparc* experiment

For the *small engines* experiment, changing the threshold had no effect. This result suggests that for any sequence X - Y - Z in the *small engines* text, if $(X_h Z_h)$ appears as a reduced pair, $(X_h Y_h)$ does not, but it is not a universal observation.



Implementation Note

For testing purposes, the NMR analyzer allows the optional storage of bracketing structures and NMR structures indexed by noun phrase number. Bracketing and NMR assignment proceed as usual, using information from preceding analyses only. But whenever the system *would have* consulted the user, it consults the indexed structures for the current phrase instead. After a text has been analyzed once, it can be reanalyzed fully automatically to test different threshold values, different “best NMR” formulae, etc.

Branching Frequencies

Ter Stal (1996) confirms earlier results from Resnik (1993) and Lauer & Dras (1994) that between 60% and 70% of noun-noun-noun compounds in text are left-branching. Section 4.3.4 suggested that the bracketer could guess left-branching when there is no confidence in right-branching. Such a guess would be justified assuming the predominance of left-branching compounds. Results from the *small engines* experiment confirm the bias for left-branching.

	<i>left-branching</i>	<i>right-branching</i>
<i>noun-noun-noun</i>	47 (96%)	2 (4%)
<i>adjective-noun-noun</i>	31 (84%)	6 (16%)
<i>total</i>	78 (91%)	8 (9%)

Table 33: Branching frequencies for the *small engines* text

For the *sparc* experiment, however, the data in Table 34 show that guessing left-branching would have produced poor results.

	<i>left-branching</i>	<i>right-branching</i>
<i>noun-noun-noun</i>	41 (55%)	33 (45%)
<i>adjective-noun-noun</i>	11 (26%)	31 (74%)
<i>total</i>	52 (45%)	64 (55%)

Table 34: Branching frequencies for the *sparc* text

The predominance of left-branching compounds is apparently not universal. If the system were modified to guess left in the absence of other evidence, there are texts (like the *sparc* text) for which the bracketer would perform poorly.

4.7.2 NMRA Evaluation

System Performance

The *sparc* experiment assigned NMRs to 766 modifier-head pairs. The system's assignment is considered *correct* when the user accepts its single suggestion *or* chooses one from among its multiple suggestions. According to this definition, the system assigned 555 NMRs (72%) correctly, with the user supplying 211 labels. For 532 of the system's correct assignments (96%) it offered a single suggestion. For the 23 multiple choice suggestions, there were an average of 3.3 suggested NMRs.

Of the 500 noun phrases, 311 were unique after morphological analysis. Assuming a similar distribution, one could extrapolate that a system learning only by accumulating analyses of complete noun phrases could perform automatically no more than 38% of the time. Even using reduced pairs, only 433 of the 766 assignments (57%) were distance 0 matches. The partial matching techniques (section 4.6.3) increased the number of correct analyses by 15%.

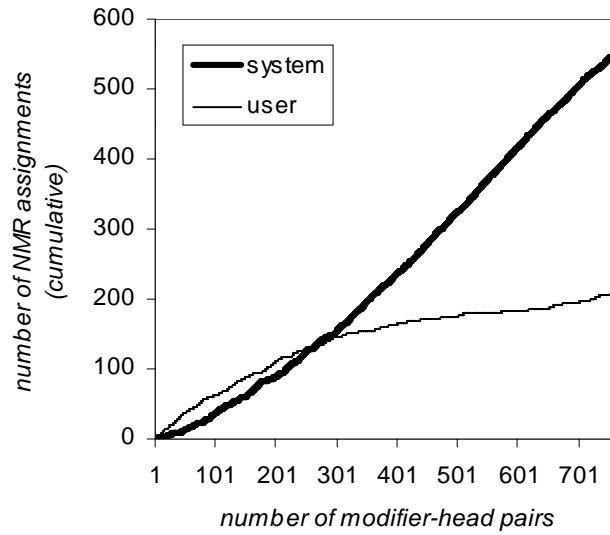


Figure 16: NMR assignments in the sparcs experiment

Figure 16 shows the cumulative number of NMR assignments supplied by the user versus those determined correctly by the system. After about 300 assignments, the system was able to make the majority of assignments automatically. The curves in the figure show that the system learns to make better suggestions as more phrases are analyzed.

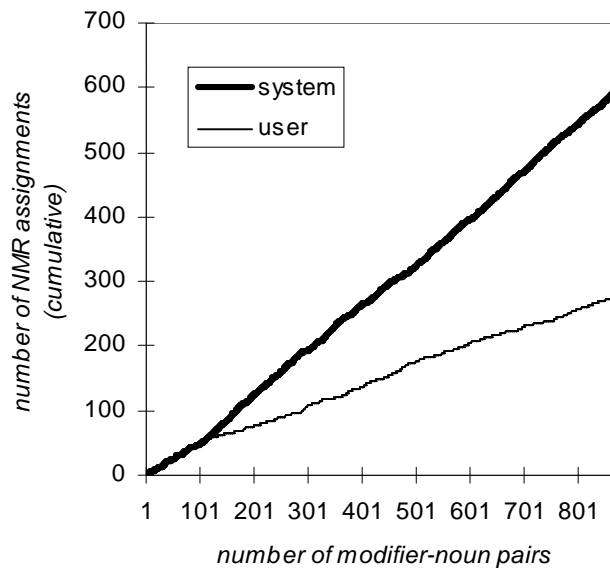


Figure 17: NMR assignments in the small engines experiment

In the *small engines* experiment (Figure 17), the system assigned NMRs to 886 modifier-noun pairs. 608 of the 886 NMRs (69%) were assigned correctly by the system. For 586 of these assignments (97.5%) the system offered a single suggestion. It had multiple suggestions (on average 3.3 again) only 22 times. There were 384 distance 0 matches (43%), meaning that partial matching increased the number of correct assignments by another 26%.

Both the *sparc* and *small engines* experiments used a version of the NMR analyzer with the *weighted relative frequency* method of choosing best NMRs from two candidate lists (see section 4.6.4). A second experiment on the *sparc* text applied five different methods to distance 3 and distance 6 triples (those that might generate two lists of candidate NMRs: \mathbf{L}_M and \mathbf{L}_H). In addition to the *absolute frequency* and *weighted relative frequency* methods from section 4.6.4, the implemented system has a *basic union* method, a *modifier preference* method and a *head preference* method.

Whereas the *absolute frequency* method takes the most frequent NMRs in $\mathbf{L}_M \cup \mathbf{L}_H$, the *basic union* method takes *all* of the NMRs in $\mathbf{L}_M \cup \mathbf{L}_H$ (without duplicates) and presents them to the. The *modifier preference* method presents the most frequent NMRs in \mathbf{L}_M only, ignoring \mathbf{L}_H . *Head preference* uses \mathbf{L}_H and ignores \mathbf{L}_M . The five methods were applied to 324 distance 3 or distance 6 triples in the *sparc* text. Results appear in Table 35.

	<i>accept</i>	<i>choose (average)</i>	<i>supply</i>
<i>basic union</i>	101	125 (2.8)	98
<i>absolute frequency</i>	168	22 (2.3)	134
<i>weighted relative</i>	174	16 (2.2)	134
<i>modifier preference</i>	182	12 (2.3)	130
<i>head preference</i>	143	21 (2.3)	160

Table 35: Applying different “best NMR” methods to *sparc*

As usual, *accept* represents the number of times a given method identified a single NMR as the best NMR and it was the correct one. *Choice* is the number of times the method identified 2 or more NMRs and the correct NMR was among them; (*average*) is the average number of best NMRs identified. The *supply* column shows the number of times the correct NMR was not among those chosen by the given method.

The results are inconclusive. The *absolute frequency* and *weighted relative* methods perform about equally well. *Modifier preference* performs better than *head preference*, though there is no theoretical reason to expect this result. It is difficult to compare the *basic union* method to the others. It resulted in far fewer user-supplied NMRs, but the number of multiple choice interactions was six times higher than for the other methods.

Coverage

As with the other HAIKU modules, NMRA is affected by the quality of DIPETT parse trees, though to a lesser degree. During the *small engines* experiment we noted that several noun phrases in sentences did not receive NMR analysis because of errors in the parses for those sentences (Barker *et al.* 1998). It would not be feasible to compare directly the number of NMR assignments made by HAIKU to the number of relationships in the text—even in a text of only a few hundred sentences like *small engines*.

To measure coverage, I sampled 100 modifier-noun pairs at random from the *small engines* text and found that 87 of them appeared in HAIKU's output. At the 95% confidence level, the system extracted between 79.0% and 92.2% of the modifier-noun relationships in the text.

User Burden

To measure the burden that NMR analysis places on the user, we assigned an onus rating to each interaction during the *small engines* experiment. As explained in section 1.4.4, the onus is an integer from 0 to 3, with 0 assigned to the simplest interactions and 3 to the most arduous. The average user onus rating was 0.1 for NMR interactions in the *small engines* experiment. 808 of the 886 NMR assignments received an onus rating of 0; 71 had a rating of 1; 7 received a rating of 2. No interactions were rated onus level 3.

4.7.3 Some Difficulties

Questionably Endocentric Compounds

Section 4.1.1 restricted NMRA to transparent endocentric compounds. In that section I also pointed out that the line between endocentric and exocentric is sometimes blurred. For example, it is unclear whether compounds such as those in (224) are endocentric or exocentric.

(224) *toy truck, teddy bear, stuffed animal, gingerbread man, fake gun, chocolate bunny*

If the compounds are taken to be endocentric they pose no problem: assign them Equative, Equative, Property, Material, Property and Material respectively. The NMR paraphrases are shown in (225).

(225) *toy is also truck*
teddy is also bear
stuffed_animal is stuffed
gingerbread_man is made of gingerbread
fake_gun is fake
chocolate_bunny is made of chocolate

This interpretation has precedence. Leonard (1984) assigns her Material label to (226). Lauer (1995a) claims that (227) and (228) are adequately interpreted as equative and material⁵ relationships, even if metaphorically.

(226) *stone lion*

(227) *barrel chest*

(228) *steel father*

Kamp (1975) takes the opposite position. He argues that compounds such as those in (225) are purely exocentric—that a *fake gun* is by definition an entity that is never a *gun*. Franks (1995) distinguishes two kinds of *privative* adjective: *negators* (such as *fake*) and *equivocators* (such as *apparent*). Negators contradict defining features of the head noun; equivocators merely undermine them.

Warren (1978) introduces a special relation Resemblance, with the paraphrase *be like*. There is no reason why such a relationship could not be added to HAIKU's list of NMRs. (229) shows the possible paraphrases for the compounds in (224).

(229) *toy_truck* is like a *truck*
teddy_bear is like a *bear*
stuffed_animal is like an *animal*
gingerbread_man is like a *man*
fake_gun is like a *gun*
chocolate_bunny is like a *bunny*

The problem with assigning Resemblance to all of these examples is that the interpretations under the existing NMRs are valid: a *chocolate bunny* really is made of *chocolate*. Where the examples fail is not in the assignment of NMRs, but in the generation of *isa* structures based on the assumption that they are purely endocentric. One solution is to allow the user to defeat generating the *isa* structure for each NMR assignment. A second solution would be to invent a new generalization structure that is weaker than *isa*. It would again be up to the user to override *isa* generation in favour of an *islikea* structure (for example).



Implementation Note

The *isa* structure generator already checks the NMR before generating *isa* structures. Property NMRs do not result in *isa* structures. New NMRs (such as Resemblance) could trivially be added to the list of non *isa*-generating NMRs. Only slightly less trivial would be the addition of an *isatype* field to each NMR: {*isa*; *islikea*; none}.

⁵ Lauer does not actually use relationship labels such as Equative and Material. He does refer to (227) as a copula compound that can be paraphrased *chest that is a barrel*. His paraphrase for (228) is *father of steel*.

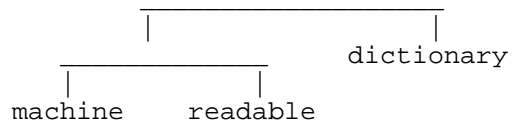
Manual Modifier Bracketing Editor:

> Please enter the bracketing for the current NP using the word numbers in order. The outermost brackets are optional.

E.g.: [1 [2 3]] 4

1 machine
2 readable
3 dictionary

|: **[1 2] 3**



> Do you accept the bracketing [r/h/m/a/Y]?
(r)edo, (h)istory off, (m)anual, (a)bort **Y**

Match type 7: (_, _, _)

For the phrase 'machine readable'
There is a relationship between
(machine) and (machine_readable).

> Please enter a valid NMR label ('a' to abort): **a**

NMR Analysis for this pair aborted by user.

Match type 7: (_, _, _)

For the phrase 'machine_readable dictionary'
There is a relationship between
(machine_readable) and (machine_readable_dictionary).

> Please enter a valid NMR label ('a' to abort): **stat**

> For the phrase 'machine_readable dictionary'
Do you accept the assignment:
State (stat): machine_readable_dictionary is in a state of
machine_readable

[n/a/<nmr>/Y] **Y**

NOUN MODIFIER RELATIONSHIPS

State (stat): machine_readable_dictionary is in a state of
machine_readable

machine_readable_dictionary isa dictionary

Figure 18: Bracketing and NMRA for a postpositive adjective

Adjectives in Paraphrases

Section 4.4.1 suggested that WordNet could be used to improve awkward paraphrases involving adjectives. The awkward paraphrase (234) of example (233) could be fixed by using the WordNet pertainym for *solar*, which is *Sun*.

(233) *solar system*

(234) *solar is located at solar system*

(235) *Sun is located at solar system*

Although pertainyms would make NMRA interaction smoother, I have chosen *not* to implement this feature. The overhead of the WordNet in Prolog API would be justified by using WordNet in the many other parts of HAIKU that could benefit from it. That system would be a very different HAIKU than the one described here. Identifying all of those parts (including pertainyms in paraphrases) and implementing a full HAIKU-with-WordNet would be a good future project (see section 5.5.2).

4.8 An Example

In this section I show the HAIKU NMRA interaction for four noun phrases, starting with no previous bracketings or analyses:

(236) *small gasoline engine*

(237) *the repair of diesel engines*

(238) *diesel engine repair shop*

(239) *an auto repair centre*

Of course, it is unlikely that these four noun phrases would occur isolated and in sequence in a text. Nonetheless, noun phrases of this kind, scattered throughout the *small engines* text, illustrate well the use of previous phrases to help bracketing and NMR assignment.

Since (236) is the first noun phrase, no previous bracketed pairs or NMR assignments can help analysis. The user has to make bracketing decisions and supply NMR labels directly. Results of the interaction are two NMR assignments and only one `isa` relationship (recall that `isa` relationships are suppressed for Property).

```
String (236)      small gasoline engine
HAIKU: Noun Modifier Relationship Analysis of current input ...
> For the phrase 'small gasoline engine'
  is that 'small gasoline' [Y/n]? n

      |-----|
      |         |
  small         |
                |-----|
                |         |
            gasoline     engine

> Do you accept the bracketing [r/h/m/a/Y]?
  (r)edo, (h)istory off, (m)anual, (a)bort Y

Match type 7: (_, _, _)

For the phrase 'gasoline engine'
There is a relationship between
  (gasoline) and (gasoline_engine).

> Please enter a valid NMR label ('a' to abort): inst

> For the phrase 'gasoline engine'
  Do you accept the assignment:
  Instrument (inst): gasoline is used in gasoline_engine

  [n/a/<nmr>/Y] Y

Match type 7: (_, _, _)

For the phrase 'small gasoline_engine'
There is a relationship between
  (small) and (small_gasoline_engine).

> Please enter a valid NMR label ('a' to abort): prop

> For the phrase 'small gasoline_engine'
  Do you accept the assignment:
  Property (prop): small_gasoline_engine is small

  [n/a/<nmr>/Y] Y

NOUN MODIFIER RELATIONSHIPS
  Property (prop): small_gasoline_engine is small
  Instrument (inst): gasoline is used in gasoline_engine

  gasoline_engine isa engine
```

Figure 19: NMRA interaction for (236)

For phrase (237) no bracketing decisions are required. The subphrase *diesel engine* has never been seen before, but *engine* has appeared as the head in two other reduced pairs: *gasoline engine* and *small engine*. Instrument and Property were each assigned once and therefore have equal weight. Both are suggested to the user in the paraphrases from section 4.4.1.

The complete phrase *repair of diesel engine* is reduced to the modifier-head-marker triple (*engine, repair, of*). The word *engine* has never been encountered as a modifier and *repair* has not occurred as a head. The preposition *of* appears in the NMR marker dictionary mapped to twelve NMRs. All twelve are suggested to the user, with *engine* and *repair* appearing in the paraphrases.

```
String (237)      the repair of diesel engines
HAIKU: Noun Modifier Relationship Analysis of current input ...
Match type 3: (diesel, _, nil) or (_, engine, nil)
[prop,inst]: 0.5
For the phrase 'diesel engine'
There is a relationship between
  (diesel) and (diesel_engine).
The NMR Analyzer's best suggestion(s) for this input:
  (1) Property (prop): diesel_engine is diesel
  (2) Instrument (inst): diesel is used in diesel_engine
> Please enter a number between 1 and 2
  or enter a valid NMR label ('a' to abort): 2
> For the phrase 'diesel engine'
  Do you accept the assignment:
  Instrument (inst): diesel is used in diesel_engine
  [n/a/<nmr>/Y] Y
Match type 5: nmrDict(of,
[agt,caus,cont,equa,led,matr,obj,poss,prop,resu,src,top])
For the phrase 'repair of diesel_engine'
There is a relationship between
  (diesel_engine) and (repair).
The NMR Analyzer's best suggestion(s) for this input:
  (1) Agent (agt): repair is performed by diesel_engine
  (2) Cause (caus): diesel_engine causes repair
  (3) Content (cont): diesel_engine is contained in repair
  (4) Equative (equa): diesel_engine is also repair
  (5) Located (led): diesel_engine is located at repair
  (6) Material (matr): repair is made of diesel_engine
  (7) Object (obj): diesel_engine is acted on by repair
  (8) Possessor (poss): diesel_engine has repair
  (9) Property (prop): repair is diesel_engine
  (10) Result (resu): diesel_engine is a result of repair
  (11) Source (src): diesel_engine is the source of repair
  (12) Topic (top): repair is concerned with diesel_engine
> Please enter a number between 1 and 12
  or enter a valid NMR label ('a' to abort): 7
> For the phrase 'repair of diesel_engine'
  Do you accept the assignment:
  Object (obj): diesel_engine is acted on by repair
  [n/a/<nmr>/Y] Y
```



```

Match type 0: (diesel, engine, nil)
Most frequent NMRs: [inst] (100.0%)

> For the phrase 'diesel engine'
Do you accept the assignment:
Instrument (inst): diesel is used in diesel_engine

  [n/a/<nmr>/Y] Y

Match type 2: (engine, repair, _)
Most frequent NMRs: [obj] (100.0%)

> For the phrase 'diesel_engine repair'
Do you accept the assignment:
Object (obj): diesel_engine is acted on by diesel_engine_repair

  [n/a/<nmr>/Y] Y

Match type 7: (_, _, _)

For the phrase 'diesel_engine_repair shop'
There is a relationship between
  (diesel_engine_repair) and (diesel_engine_repair_shop).

> Please enter a valid NMR label ('a' to abort): purp

> For the phrase 'diesel_engine_repair shop'
Do you accept the assignment:
Purpose (purp): diesel_engine_repair_shop is meant for
                 diesel_engine_repair

  [n/a/<nmr>/Y] Y

NOUN MODIFIER RELATIONSHIPS
Purpose (purp): diesel_engine_repair_shop is meant for
                 diesel_engine_repair
Object (obj): diesel_engine is acted on by diesel_engine_repair
Instrument (inst): diesel is used in diesel_engine

diesel_engine_repair_shop isa shop
diesel_engine_repair isa repair
diesel_engine isa engine

```

Figure 21: NMRA interaction for (238)

The bracketer must ask the user to supply the branching decision for (239), since neither *auto repair* nor *auto centre* have appeared in bracketing before. For NMR assignment, *auto* has never appeared as a modifier, but *repair* has appeared twice as a head. Both times Object was assigned. For *auto repair centre*, *repair* has appeared once as a modifier (as Purpose in *diesel engine repair shop*); *centre* has never been encountered as a head. Purpose is suggested to the user.

previous triples. It chooses one or more of these candidates as the most appropriate for the current triple using their weighted relative frequency. For each NMR assignment to a premodifier-head compound, HAIKU also generates an *isa* structure. In doing so it assumes that the compound is endocentric.

Performance evaluations of the bracketer showed that the system's reliance on the user for branching decisions decreases as more noun phrases are analyzed. In the *sparc* experiment, adjusting the branching frequency threshold had the expected effect: higher thresholds made the bracketer more conservative in using previous bracketing evidence. This resulted in higher accuracy, but fewer decisions were attempted by the system. In the *small engines* experiment, changing the threshold had no effect.

By the end of each experiment, HAIKU was assigning the majority of NMRs automatically, having learned from previous assignments. The system recovered between 79% and 92% of the relationships in the *small engines* text (with 95% confidence). The burden placed on the user by NMRA was low, with an average onus of 0.1 for interactions in the *small engines* experiment. No new NMRs were needed in either the *sparc* experiment or the *small engines* experiment.