Appears in Computational Natural Language Learning (CoNLL-99), pages 43-52.
A workshop at the 9th Conf. of the European Chapter of the Assoc. for Computational Linguistics (EACL-99).
Bergen, Norway, June, 1999.

Learning Transformation Rules to Find Grammatical Relations*

Lisa Ferro and **Marc Vilain** and **Alexander Yeh**
The MITRE Corporation
202 Burlington Rd.
Bedford, MA 01730
USA
{lferro,mbv,asy}@mitre.org

Abstract

Grammatical relationships are an important level of natural language processing. We present a trainable approach to find these relationships through transformation sequences and error-driven learning. Our approach finds grammatical relationships between core syntax groups and bypasses much of the parsing phase. On our training and test set, our procedure achieves 63.6% recall and 77.3% precision (f-score = 69.8).

Introduction

An important level of natural language processing is the finding of grammatical relationships such as subject, object, modifier, etc. Such relationships are the objects of study in relational grammar [Perlmutter, 1983]. Many systems (e.g., the KERNEL system [Palmer *et al.*, 1993]) use these relationships as an intermediate form when determining the semantics of syntactically parsed text. In the SPARKLE project [Carroll *et al.*, 1997a], grammatical relations form the layer above the phrasal-level in a three layer syntax scheme. Grammatical relationships are often stored in some type of structure like the F-structures of lexical-functional grammar [Kaplan, 1994].

Our own interest in grammatical relations is as a semantic basis for information extraction in the *Alembic* system. The extraction approach we are currently investigating exploits grammatical relations as an intermediary between surface syntactic phrases and propositional semantic interpretations. By directly associating syntactic heads with their arguments and modifiers, we are hoping that these grammatical relations will provide a high degree of generality and reliability to the process of composing semantic representations. This ability to "parse" into a semantic representation is according to Charniak [Charniak, 1997, p. 42], "the most important task to be tackled now."

In this paper, we describe a system to learn rules for finding grammatical relationships when just given a partial parse with entities like names, core noun and verb phrases (noun and verb groups) and semi-accurate estimates of the attachments of prepositions and subordinate conjunctions. In our system, the different entities, attachments and relationships are found using rule sequence processors that are cascaded together. Each processor can be thought of as approximating some aspect of the underlying grammar by finite-state transduction.

We present the problem scope of interest to us, as well as the data annotations required to support our investigation. We also present a decision procedure for finding grammatical relationships. In brief, on our training and test set, our procedure achieves 63.6% recall and 77.3% precision, for an f-score of 69.8.

Phrase Structure and Grammatical Relations

In standard derivational approaches to syntax, starting as early as 1965 [Chomsky, 1965], the notion of grammatical relationship is typically parasitic on that of phrase structure. That is to say, the primary vehicles of syntactic analysis are phrase structure trees; grammatical relationships, if they are to be considered at all, are given as a secondary analysis defined in terms of phrase structure. The surface subject of a sentence, for example, is thus no more than the NP attached by the production S $\rightarrow$ NP VP; i.e., it is the left-most NP daughter of an S node.

The present paper takes an alternate outlook. In our current work, grammatical relationships play a central role, to the extent even of replacing phrase structure as the descriptive vehicle for many syntactic phenomena. To be specific, our approach to syntax operates at two levels: (1) that of core phrases, which are an-

* This paper reports on work performed at the MITRE Corporation under the support of the MITRE Sponsored Research Program. Helpful assistance has been given by Yuval Krymolowski, Lynette Hirschman and an anonymous reviewer.

alyzed through standard derivational syntax, and (2) that of argument and modifier attachments, which are analyzed through grammatical relations. These two levels roughly correspond to the top and bottom layers of the three layer syntax annotation scheme in the SPARKLE project [Carroll *et al.*, 1997a].

Core syntactic phrases

In recent years, a consensus of sorts has emerged that postulates some core level of phrase analysis. By this we mean the kind of non-recursive simplifications of the NP and VP that in the literature go by names such as noun/verb groups [Appelt *et al.*, 1993], chunks [Abney, 1996], or base NPs [Ramshaw and Marcus, 1995].

The common thread between these approaches and ours is to approximate full noun phrases or verb phrases by only parsing their non-recursive core, and thus not attaching modifiers or arguments. For English noun phrases, this amounts to roughly the span between the determiner and the head noun; for English verb phrases, the span runs roughly from the auxiliary to the head verb. We call such simplified syntactic categories *groups*, and consider in particular, noun, verb, adverb, adjective, and *IN* groups.[1] An *IN* group[2] contains a preposition or subordinate conjunction (including *wh*-words and "that").

For example, for *"I saw the cat that ran."*, we have the following core phrase analysis:

$$[\text{I}]_{ng}\ [\text{saw}]_{vg}\ [\text{the cat}]_{ng}\ [\text{that}]_{ig}\ [\text{ran}]_{vg}.$$

where $[...]_{ng}$ indicates a noun group, $[...]_{vg}$ a verb group, and $[...]_{ig}$ an IN group.

In English and other languages where core phrases (groups) can be analyzed by head-out (island-like) parsing, the group head-words are basically a by-product of the core phrase analysis.

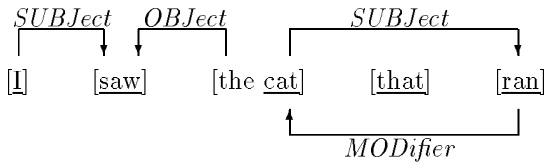
Distinguishing core syntax groups from traditional syntactic phrases (such as NPs) is of interest because it singles out what is usually thought of as easy to parse, and allows that piece of the parsing problem to be addressed by such comparatively simple means as finite-state machines or transformation sequences. What is then left of the parsing problem is the difficult stuff: namely the attachment of prepositional phrases, relative clauses, and other constructs that serve in modification, adjunctive, or argument-passing roles.

[1] In addition, for the noun group, our definition encompasses the named entity task, familiar from information extraction [Def, 1995]. Named entities include among others the names of people, places, and organizations, as well as dates, expressions of money, and (in an idiosyncratic extension) titles, job descriptions, and honorifics.

[2] The name comes from the Penn Treebank part-of-speech label for prepositions and subordinate conjunctions.

Grammatical relations

In the present work, we encode this hard stuff through a small repertoire of *grammatical relations*. These relations hold directly between constituents, and as such define a graph, with core constituents as nodes in the graph, and relations as labeled arcs. Our previous example, for instance, generates the following grammatical relations graph (head words underlined):



Our grammatical relations effectively replace the recursive $\overline{X}$ analysis of traditional phrase structure grammar. In this respect, the approach bears resemblance to a dependency grammar, in that it has no notion of a spanning S node, or of intermediate constituents corresponding to argument and modifier attachments.

One major point of departure from dependency grammar, however, is that these grammatical relation graphs can generally not be reduced to labeled trees. This happens as a result of argument passing, as in

[Fred] [promised] [to help] [John]

where *[Fred]* is both the subject of *[promised]* and *[to help]*. This also happens as a result of argument-modifier cycles, as in

[I] [saw] [the cat] [that] [ran]

where the relationships between *[the cat]* and *[ran]* form a cycle: *[the cat]* has a subject relationship/dependency to *[ran]*, and *[ran]* has a modifier dependency to *[the cat]*, since *[ran]* helps indicate (modifies) which cat is seen.

There has been some work at making additions to extract grammatical relationships from a dependency tree structure [Bröker, 1998, Lai and Huang, 1998] so that one first produces a surface structure dependency tree with a syntactic parse and then extracts grammatical relationships from that tree. In contrast, we skip trying to find a surface structure tree and just proceed to more directly finding the grammatical relationships, which are the relationships of interest to us.

A reason for skipping the tree stage is that extracting grammatical relations from a surface structure tree is often a nontrivial task by itself. For instance, the precise relationship holding between two constituents in a surface structure tree cannot be derived unambiguously from their relative attachments. Contrast, for example *"the attack on the military base"* with *"the attack on March 24"*. Both of these have the same underlying surface structure (a PP attached to an NP), but the former encodes the direct object of a verb nominalization, while the latter encodes a time modifier. Also, in a surface structure tree, long-distance dependencies between heads and arguments are not explicitly indicated by attachments between the appropriate parts of the text. For instance in *"Fred promised to help John"*, no direct attachment exists between the *"Fred"* in the text and the *"help"* in the text, despite the fact that the former is the subject of the latter.

For our purposes, we have delineated approximately a dozen head-to-argument relationships as well as a commensurate number of modification relationships. Among the head-to-argument relationships, we have the deep subject and object (SUBJ and OBJ respectively), and also include the surface subject and object of copulas (COP-SUBJ and the various COP-OBJ forms). In addition, we include a number of relationships (e.g., PP-SUBJ, PP-OBJ) for arguments that are mediated by prepositional phrases. An example is in

PP-OBJect *OBJect*

[the attack] [on] [the military base]

where *[the attack]*, a noun group with a verb nominalization, has its object *[the military base]* passed to it via the preposition in *[on]*. Among modifier relationships, we designate both generic modification and some specializations like locational and temporal modification. A complete definition of all the grammatical relations is beyond the scope of this paper, but we give a summary of usage in Table 1. An earlier version of the definitions can be found in our annotation guidelines [Ferro, 1998]. The appendix shows some examples of grammatical relationship labeling from our experiments.

Our set of relationships is similar to the set used in the SPARKLE project [Carroll *et al.*, 1997a] [Carroll *et al.*, 1998a]. One difference is that we make many semantically-based distinctions between what SPARKLE calls a **mod**ifier, such as time and location modifiers, and the various arguments of event nouns.

Semantic interpretation

A major motivation for this approach is that it supports a direct mapping into semantic interpretations. In our framework, semantic interpretations are given in a neo-Davidsonian propositional logic. Grammatical relations are thus interpreted in terms of mappings and relationships between the constants and variables of the propositional language. For instance, the deep subject relation (SUBJ) maps to the first position of a predicate's argument list, the deep object (OBJ) to the second such position, and so forth.

Our example sentence, *"I saw the cat that ran"* thus translates directly to the following:

Proposition	Comment
saw(x1 x2)	SUBJ and OBJ relations
I(x1)	
cat(x2)	
ran(x2)=e3	SUBJ relation (e3 is the event variable)
mod(e3 x2)	MOD relation

We do not have an explicit level for clauses between our core phrase and grammatical relations levels. However, we do have a set of implicit clauses in that each verb (event) and its arguments can be deemed a base level clause. In our example *"I saw the cat that ran"*, we have two such base level clauses. *"saw"* and its arguments form the clause *"I saw the cat"*. *"ran"* and its argument form the clause *"the cat ran"*. Each noun with a possible semantic class of "act" or "process" in Wordnet [Miller, 1990] (and that noun's arguments) can likewise be deemed a base level clause.

The Processing Model

Our system uses transformation-based error-driven learning to automatically learn rules from training examples [Brill and Resnik, 1994].

One first runs the system on a training set, which starts with no grammatical relations marked. This training run moves in iterations, with each iteration producing the next rule that yields the best net gain in the training set (number of matching relationships found minus the number of spurious relationships introduced). On ties, rules with less conditions are favored over rules with more conditions. The training run ends when the next rule found produces a net gain below a given threshold.

The rules are then run in the same order on the test set to see how well they do.

The rules are condition/action pairs that are tried on each syntax group. The actions in our system are limited to attaching (or unattaching) a relationship of a particular type from the group under consideration to that group's neighbor a certain number of groups away in a particular direction (left or right). A sample action would be to attach a SUBJ relation from the group under consideration to the group two groups away to the right.

A rule only applies to a syntax group when that group and its neighbors meet the rule's conditions. Each condition tests the group in a particular position relative to the group under consideration (e.g., two groups away to the left). All tests can be negated. Table 2 shows the possible tests.

A sample rule is when a noun group n's

- immediate group to the right has some form of the verb *"be"* as the head-word,

RELATION		EXAMPLE(s) in the format
Name	Description	*[source]* → *[target]* in "text"
subj	subject — subject of a verb — link a copula subject and object — link a state with the item in that state — link a place with the item moving to or from that place	*[I]* → *[promised]* in "I promised to help" *[I]* → *[to help]* in "I promised to help" *[the cat]* → *[ran]* in "the cat that ran" *[You]* → *[happy]* in "You are happy" *[You]* → *[a runner]* in "You are a runner" *[you]* → *[happy]* in "They made you happy" *[I]* → *[home]* in "I went home"
obj	object — object of a verb — object of an adjective — surface subject in passives — object of a preposition, not for partitives or subsets — object of an adverbial clause complementizer	*[saw]* ← *[the cat]* in "I saw the cat" *[promised]* ← *[to help]* in "I promised to help you" *[happy]* ← *[to help]* in "I was happy to help" *[I]* → *[was seen]* in "I was seen by a cat" *[by]* ← *[the tree]* in "I was by the tree" *[After]* ← *[left]* in "After I left, I ate"
loc-obj	location object –link a movement verb with a place where entities are moving to or from	*[went]* ← *[home]* in "I went home" *[went]* ← *[in]* in "I went in the house
indobj	indirect object	*[gave]* ← *[you]* in "I gave you a cake"
empty	use instead of "subj" relation when subject is an expletive (existential) "it" or "there"	*[There]* → *[trees]* in "There are trees"
pp-subj	genitive functional "of"'s use instead of "subj" relation when the subject is linked via a preposition, links preposition to its head	*[name]* ← *[of]* in "name of the building" *[was seen]* ← *[by]* in "I was seen by a cat"
pp-obj	nongenitive functional "of"'s use in place of "obj" relation when the object is linked via a preposition, links preposition to its head	*[age]* ← *[of]* in "age of 12" *[the attack]* ← *[on]* in "the attack on the base"
pp-io	use in place of "indobj" relation when the indirect object is linked via a preposition, links preposition to its head	*[gave]* ← *[to]* in "gave a cake to them"
cop-subj	surface subject for a copula	*[You]* → *[are]* in "You are happy"
n-cop-obj	surface nominative object for a copula	*[is]* ← *[a rock]* in "It is a rock"
p-cop-obj	surface predicate object for a copula	*[are]* ← *[happy]* in "You are happy"
subset	subset	*[five]* → *[the kids]* in "five of the kids"
mod	generic modifier (use when modifier does not fit in a case below)	*[the cat]* ← *[ran]* in "the cat that ran" *[ran]* ← *[with]* in "I ran with new shoes"
mod-loc	location modifier	*[ate]* ← *[at]* in "I ate at home"
mod-time	time modifier	*[ate]* ← *[at]* in "I ate at midnight" *[Yesterday]* → *[ate]* in "Yesterday, I ate"
mod-poss	possessive modifier	*[the cat]* → *[toy]* in "the cat's toy"
mod-quant	quantity modifier (partitive)	*[hundreds]* → *[people]* in "hundreds of people"
mod-ident	identity modifier (names)	*[a cat]* ← *[Fuzzy]* in "a cat named Fuzzy" *[the winner]* ← *[Pat Kay]* in "the winner, Pat Kay, is"
mod-scalar	scalar modifier	*[16 years]* → *[ago]* in "16 years ago"

Table 1: Summary of grammatical relationships

Test Type	Example, Sample Value(s)
group type	noun, verb
verb group property	passive, infinitival, unconjugated present participle
end group in a sentence	first, last
pp-attachment	Is a preposition or subordinate conjunction attached to the group under consideration?
group contains	a particular lexeme or part-of-speech
between two groups, there is	a particular lexeme or part-of-speech
group's head (main) word	"cat"
head word part-of-speech	common plural noun
head word within a named entity	person, organization
head word subcategorization and complement categories (from Comlex [Wolff *et al.*, 1995], over 100 categories)	intransitive verbs
head word semantic classes (from Wordnet [Miller, 1990], 25 noun and 15 verb classes)	process, communication
punctuation or coordinating conjunction	exist between two groups?
head word in a word list?	list of relative pronouns, list of partitive quantities (e.g., *"some"*)

Table 2: Possible tests

- immediate group to the left is not an IN group (preposition, *wh*-word, etc.) and
- n's head-word is not an existential *"there"*

make n a SUBJ of the group two groups over to n's right.

When applied to the group *[The cat]* (head words are underlined) in the sentence

[The cat] [was] [very happy].

this rule makes *[The cat]* a SUBJect of *[very happy]*.

Searching over the space of possible rules is very computationally expensive. Our system has features to make it easier to perform searching in parallel and to minimize the amount of work that needs to be undone once a rule is selected. With these features, rules that (un)attach different types of relationships or relationships at different distances can be searched independently of each other in parallel.

One feature is that the action of any rule only affects the applicability of rules with either the exact same or opposite action. For example, selecting and running a rule which attaches a MOD relationship to the group that is two groups to the right only can affect the applicability of other rules that either attach or unattach a MOD relationship to the group that is two groups to the right.

Another feature is the use of net gain as a proxy measure during training. The actual measure by which we judge the system's performance is called an *f-score*. This *f-score* is a type of harmonic mean of the precision (p) and recall (r) and is given by $2pr/(p+r)$. Unfortunately, this measure is nonlinear, and the application of a new rule can alter the effects of all other possible rules on the *f-score*. To enable the described parallel search to take place, we need a measure in which how a rule affects that measure only depends on other rules with either the exact same or opposite action. The net gain measure has this trait, so we use it as a proxy for the *f-score* during training.

Another way to increase the learning speed is to restrict the number of possible combinations of conditions/constraints or actions to search over. Each rule is automatically limited to only considering one type of syntactic group. Then when searching over possible conditions to add to that rule, the system only needs to consider the parts-of-speech, semantic classes, etc. applicable to that type of group.

Many other restrictions are possible. One can estimate which restrictions to try by making some training and test runs with preliminary data sets and seeing what restrictions seem to have no effect on performance, etc. The restrictions used in our experiments are described below.

Experiments

The Data

Our data consists of bodies of some elementary school reading comprehension tests. For our purposes, these tests have the advantage of having a fairly predictable size (each body has about 100 relationships and syntax groups) and a consistent style of writing. The tests are also on a wide range of topics, so we avoid a narrow specialized vocabulary. Our training set has 1963 relationships (2153 syntax groups, 3299 words) and our test set has 748 relationships (830 syntax groups, 1151 words).

We prepared the data by first manually removing the headers and the questions at the end for each test. We then manually annotated the remainder for named entities, syntax groups and relationships. As the system reads in our data, it automatically breaks the data into lexemes and sentences, tags the lexemes for part-of-speech and estimates the attachments of prepositions and subordinate conjunctions. The part-of-speech tagging uses a high-performance tagger based on [Brill, 1993]. The attachment estimation uses a procedure described in [Yeh and Vilain, 1998] when multiple left attachment possibilities exist and four simple rules when no or only one left attachment possibility exists. Previous testing indicates that the estimation procedure is about 75% accurate.

Parameter Settings for Training

As described earlier, a training run uses many parameter settings. Examples include where to look for relationships and to test conditions, the maximum number of constraints allowed in a rule, etc.

Based on the observation that 95% of the relationships are to at most three groups away in the training set, we decided to limit the search for relationships to at most three groups in length. To keep the number of possible constraints down, we disallowed the negations of most tests for the presence of a particular lexeme or lexeme stem.

To help determine many of the settings, we made some preliminary runs using different subsets of our final training set as the preliminary training and test sets. This kept the final test set unexamined during development. From these preliminary runs, we decided to limit a rule to at most three constraints[3] in order to keep the training time reasonable. We found a number of limitations that help speed up training and seemed to have no effect on the preliminary test runs. A threshold of four was set to end a training run. So training ends when it can no longer find a rule that produces at least a net gain of four in the score. Only syntax groups spanned by the relationship being attached or unattached and those groups' immediate neighbors were allowed to be mentioned in a rule's conditions. Each condition testing a head-word had to test a head-word of a different group. Except for the lexemes "of", "?" and a few determiners like *"the"*, tests for single lexemes were removed. Also disallowed were negations of tests for the presence of a particular part-of-speech anywhere within a syntax group.

In our preliminary runs, lowering the threshold tended to raise recall and lower precision.

The Results

Training produced a sequence of 95 rules which had 63.6% recall and 77.3% precision for an f-score of 69.8 when run on the test set. In our test set, the key relationships, SUBJ and OBJ, formed the bulk of the relationships (61%). Both recall and precision for both SUBJ and OBJ were above 70%, which pleased us. Because of their relative abundance in the test set, these two relationships also had the most number of errors in absolute terms. Combined, the two accounted for 45% of the recall errors and 66% of the precision errors. In terms of percentages, recall was low for many of the less common relationships, such as generic, time and location *modification* relationships. In addition, the relative precision was low for those modification relationships. The appendix shows some examples of our system responding to the test set.

To see how well the rules, which were trained on reading comprehension test bodies, would carry over to other texts of non-specialized domains, we examined a set of six broadcast news stories. This set had 525 relationships (585 syntax groups, 1129 words). By some measures, this set was fairly similar to our training and test sets. In all three sets, 33–34% of the relationships were OBJ and 26–28% were SUBJ. The broadcast news set did tend to have relationships between groups that were slightly further apart:

	Percent of Relations with Length		
Set	≤ 1	≤ 2	≤ 3
training	66%	87%	95%
test	68%	89%	96%
broadcast news	65%	84%	90%

This tendency, plus differences in the relative proportions of various modification relationships are probably what produced the drop in results when we tested the rules against this news set: recall at 54.6%, precision at 70.5% (f-score at 61.6%).

To estimate how fast the results would improve by adding more training data, we had the system learn rules on a new smaller training set and then tested

[3] In addition to the constraint on the relationship's source group type.

against the regular test set. Recall dropped to 57.8%, precision to 76.2%. The smaller training set had 981 relationships (50% of the original training set). So doubling the training data here (going from the smaller to the regular training set) reduced the smaller training set's recall error of 42.2% by 14% and the precision error of 23.8% by 5%. Using the broadcast news set as a test produced similar error reduction results.

One complication of our current scoring scheme is that identifying a *modification* relationship and mistyping it is more harshly penalized than not finding a *modification* relationship at all. For example, finding a *modification* relationship, but mistakingly calling it a generic modifier instead of a time modifier produces both a missed key error (not finding a time modifier) and a spurious response error (responding with a generic modifier where none exists). Not finding that *modification* relationship at all just produces a missed key error (not finding a time modifier). This complication, coupled with the fact that generic, time and location modifiers often have a similar surface appearance (all are often headed by a preposition or a complementizer) may have been responsible for the low recall and precision scores for these types of modifiers. Even the training scores for these types of modifiers were particularly low. To test how well our system finds these three types of *modification* when one does not care about specifying the sub-type, we reran the original training and test with the three sub-types merged into one sub-type in the annotation. With the merging, recall of these *modification* relationships jumped from 27.8% to 48.9%. Precision rose from 52.1% to 67.7%. Since these *modification* relationships are only about 20% of all the relationships, the overall improvement is more modest. Recall rises to 67.7%, precision to 78.6% (f-score to 72.6).

Taking this one step further, the LOC-OBJ and various PP-x arguments also all have both a low recall (below 35%) in the test and a similar surface structure to that of generic, time and location modifiers. When these argument types were merged with the three modifier types into one combined type, their combined recall was 60.4% and precision was 81.1%. The corresponding overall test recall and precision were 70.7% and 80.5%, respectively.

Comparison with Other Work

At one level, computing grammatical relationships can be seen as a parsing task, and the question naturally arises as to how well this approach compares to current state-of-the-art parsers. Direct performance comparisons, however, are elusive, since parsers are evaluated on an incommensurate tree bracketing task. For example, the SPARKLE project [Carroll *et al.*, 1997a] puts tree bracketing and grammatical relations in two different layers of syntax. Even if we disregard the questionable aspects of comparing tree bracketing apples to grammatical relation oranges, an additional complication is the fact that our approach divides the parsing task into an easy piece (core phrase boundaries) and a hard one (grammatical relations). The results we have presented here are given solely for this harder part, which may explain why at roughly 70 points of f-score, they are lower than those reported for current state-of-the-art parsers (e.g., Collins [Collins, 1997]).

More comparable to our approach are some other grammatical relation finders. Some examples for English include the English parser used in the SPARKLE project [Briscoe *et al.*,] [Carroll *et al.*, 1997b] [Carroll *et al.*, 1998b] and the finder built with a memory-based approach [Argamon *et al.*, 1998]. These relation finders make use of large annotated training data sets and/or manually generated grammars and rules. Both techniques take much effort and time. At first glance both of these finders perform better than our approach. Except for the *object* precision score of 77% in [Argamon *et al.*, 1998], both finders have grammatical relation recall and precision scores in the 80s. But a closer examination reveals that these results are not quite comparable with ours.

1. Each system is recovering a different variation of grammatical relations. As mentioned earlier, one difference between us and the SPARKLE project is that the latter ignores many of distinctions that we make for different types of modifiers. The system in [Argamon *et al.*, 1998] only finds a subset of the surface subjects and objects.

2. In addition, the evaluations of these two finders produced more complications. In an illustration of the time consuming nature of annotating or reannotating a large corpus, the SPARKLE project originally did not have time to annotate the English test data for modifier relationships. As a result, the SPARKLE English parser was originally not evaluated on how well it found modifier relationships [Carroll *et al.*, 1997b] [Carroll *et al.*, 1998b]. The reported results as of 1998 only apply to the argument (subject, object, etc.) relationships. Later on, a test corpus with modifier relationship annotation was produced. Testing the parser against this corpus produced generally lower results, with an overall recall, precision and f-score of 75% [Carroll *et al.*, 1999]. This is still better than our f-score of 70%, but not by nearly as much. This comparison ignores the fact that the results are for different versions of grammat-

ical relationships and for different test corpora.

The figures given above were the original (1998) results for the system in [Argamon *et al.*, 1998], which came from training and testing on data derived from the Penn Treebank corpus [Marcus *et al.*, 1993] in which the added null elements (like null subjects) were left in. These null elements, which were given a `-NONE-` part-of-speech, do not appear in raw text. Later (1999 results), the system was re-evaluated on the data with the added null elements removed. The *subject* results declined a little. The *object* results declined more, with the precision now lower than ours (73.6% versus 80.3%) and the f-score not much higher (80.6% versus 77.8%). This comparison is also between results with different test corpora and slightly different notions of what an *object* is.

Summary, Discussion, and Speculation

In this paper, we have presented a system for finding grammatical relationships that operates on easy-to-find constructs like noun groups. The approach is guided by a variety of knowledge sources, such as readily available lexica[4], and relies to some degree on well-understood computational infrastructure: a p-o-s tagger and an attachment procedure for preposition and subordinate conjunctions. In sample text, our system achieves 63.6% recall and 77.3% precision (f-score = 69.8) on our repertory of grammatical relationships.

This work is admittedly still in relatively early stages. Our training and test corpora, for instance, are less-than-gargantuan compared to such collections as the Penn Treebank [Marcus *et al.*, 1993]. However, the fact that we have obtained an f-score of 70 from such sparse training materials is encouraging. The recent implementation of rapid annotation tools should speed up further annotation of our own native corpus.

Another task that awaits us is a careful measurement of interannotator agreement on our version the grammatical relationships.

We are also keenly interested in applying a wider range of learning procedures to the task of identifying these grammatical relations. Indeed, a fine-grained analysis of our development test data has identified some recurring errors related to the rule sequence approach. A hypothesis for further experimentation is that these errors might productively be addressed by revisiting the way we exploit and learn rule sequences, or by some hybrid approach blending rules and statistical computations. In addition, since generic, time and location modifiers, and LOC-OBJ and various PP-x arguments often have a similar surface appearance, one might first just try to locate all such entities and then in a later phase try to classify them by type.

Different applications will need to deal with different styles of text (e.g., journalistic text versus narratives) and different standards of grammatical relationships. An additional item of experimentation is to use our system to adapt other systems, including earlier versions of our system, to these differing styles and standards.

Like other Brill transformation rule systems [Brill and Resnik, 1994], our system can take in the output of another system and try to improve on it. This suggests a relatively low expense method to adapt a hard-to-alter system that performs well on a slightly different style or standard. Our training approach accepts as a starting point an initial labeling of the data. So far, we have used an empty labeling. However, our system could just as easily start from a labeling produced as the output of the hard-to-alter system. The learning would then not be reducing the error between an empty labeling and the key annotations, but between the hard-to-alter system's output and the key annotations. By using our system in this post-processing manner, we could use a relatively small retraining set to adapt, for example, the SPARKLE English parser, to our standard of grammatical relationships without having reengineer that parser. Palmer [Palmer, 1997] used a similar approach to improve on existing word segmenters for Chinese. Trying this suggestion out is also something for us to do.

This discussion of training set size brings up perhaps the most obvious possible improvement. Namely, enlarging our very small training set. As has been mentioned, we have recently improved our annotation environment and look forward to working with more data.

Clearly we have many experiments ahead of us. But we believe that the results obtained so far are a promising start, and the potential rewards of the approach are very significant indeed.

Appendix: Examples from Test Results

Figure 1 shows some example sentences from the test results of our main experiment.[5] @ marks the relationship that our system missed. * marks the relationship that our system wrongly hypothesized. In these examples, our system handled a number of phenomena correctly, including:

- The coordination conjunction of the objects *[cars] and [trucks]*

[4] Resources to find a word's possible stem(s), semantic class(es) and subcategorization category(ies).

[5] The material came from level 2 of "The 5 W's" written by Linda Miller. It is available from Remedia Publications, 10135 E. Via Linda #D124, Scottsdale, AZ 85258, USA.

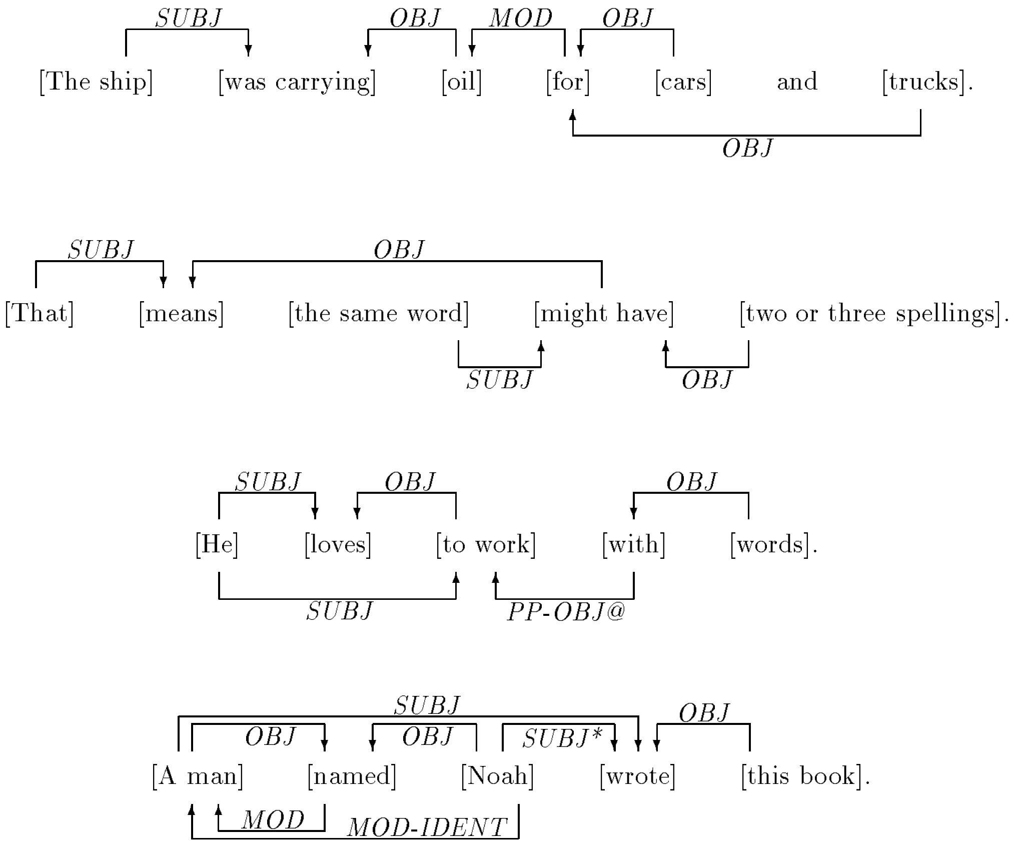


Figure 1: Example test responses from our system. @ marks the missed key. * marks the spurious response.

- The verb group *[might have]* being an object of another verb.
- The noun group *[He]* being the subject of two verbs.
- The relationships within the reduced relative clause *[A man] [named] [Noah]*, which makes one noun group a name or label for another noun group.

Our system misses a PP-OBJ relationship, which is a low occurrence relationship. Our system also accidentally make both *[A man]* and *[Noah]* subjects of the group *[wrote]* when only the former should be.

References

[Abney, 1996] S. Abney. Partial parsing via finite-state cascades. In *Proc. of ESSLI96 Workshop on Robust Parsing*, 1996.

[Appelt *et al.*, 1993] D. Appelt, J. Hobbs, J. Bear, D. Israel, and M. Tyson. Fastus: A finite-state processor for information extraction. In *13th Intl. Conf. On Artificial Intelligence (IJCAI-93)*, 1993.

[Argamon *et al.*, 1998] S. Argamon, I. Dagan, and Y. Krymolowski. A memory-based approach to learning shallow natural language patterns. In *COLING-ACL'98*, pages 67–73, Montréal, Canada, 1998. An expanded 1999 version will appear in JETAI.

[Brill and Resnik, 1994] E. Brill and P. Resnik. A rule-based approach to prepositional phrase attachment disambiguation. In *15th International Conf. on Computational Linguistics (COLING)*, 1994.

[Brill, 1993] E. Brill. *A Corpus-based Approach to Language Learning.* PhD thesis, U. Pennsylvania, 1993.

[Briscoe *et al.*,] T. Briscoe, J. Carroll, G. Carroll, S. Federici, G. Grefenstette, S. Montemagni, V. Pirrelli, I. Prodanof, M. Rooth, and M. Vannocchi. Phrasal parser software – deliverable 3.1. Available at `http://www.ilc.pi.cnr.it/-sparkle/sparkle.htm`.

[Bröker, 1998] N. Bröker. Separating surface order and syntactic relations in a dependency grammar. In *COLING-ACL'98*, pages 174–180, Montréal, Canada, 1998.

[Carroll *et al.*, 1997a] J. Carroll, T. Briscoe, N. Calzolari, S. Federici, S. Montemagni, V. Pirrelli, G. Grefenstette, A. Sanfilippo, G. Carroll, and M. Rooth. Sparkle work package 1, specification of phrasal parsing, final report. Available at `http://www.ilc.pi.cnr.it/-sparkle/sparkle.htm`, November 1997.

[Carroll *et al.*, 1997b] J. Carroll, T. Briscoe, G. Carroll, M. Light, D. Prescher, M. Rooth, S. Federici, S. Montemagni, V. Pirrelli, I. Prodanof, and M. Vannocchi. Sparkle work package 3, phrasal parsing software, deliverable d3.2. Available at `http://www.ilc.pi.cnr.it/-sparkle/sparkle.htm`, November 1997.

[Carroll *et al.*, 1998a] J. Carroll, T. Briscoe, and A. Sanfilippo. Parser evaluation: a survey and a new proposal. In *1st Intl. Conf. on Language Resources and Evaluation (LREC)*, pages 447–454, Granada, Spain, 1998.

[Carroll *et al.*, 1998b] J. Carroll, G. Minnen, and T. Briscoe. Can subcategorisation probabilities help a statistical parser? In *6th ACL/SIGDAT workshop on Very Large Corpora*, Montréal, Canada, 1998.

[Carroll *et al.*, 1999] J. Carroll, G. Minnen, and T. Briscoe. Corpus annotation for parser evaluation. In *To appear in the EACL99 workshop on Linguistically Interpreted Corpora (LINC'99)*, 1999.

[Charniak, 1997] E. Charniak. Statistical techniques for natural language parsing. *AI magazine*, 18(4):33–43, 1997.

[Chomsky, 1965] N. Chomsky. *Aspects of the Theory of Syntax.* Massachusetts Institute of Technology, 1965.

[Collins, 1997] M. Collins. Three generative, lexicalized models for statistical parsing. In *Proceedings of ACL/EACL97*, 1997.

[Def, 1995] Defense Advanced Research Projects Agency. *Proc. 6th Message Understanding Conference (MUC-6)*, November 1995.

[Ferro, 1998] L. Ferro. Guidelines for annotating grammatical relations. Unpublished annotation guidelines, 1998.

[Kaplan, 1994] R. Kaplan. The formal architecture of lexical-functional grammar. In M. Dalrymple, R. Kaplan, J. Maxwell III, and A. Zaenen, editors, *Formal issues in lexical-functional grammar.* Stanford University, 1994.

[Lai and Huang, 1998] T. B.Y. Lai and C. Huang. Complements and adjuncts in dependency grammar parsing emulated by a constrained context-free grammar. In *COLING-ACL'98 Workshop: Processing of Dependency-based Grammars*, Montréal, Canada, 1998.

[Marcus *et al.*, 1993] M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2), 1993.

[Miller, 1990] G. Miller. Wordnet: an on-line lexical database. *Intl. J. of Lexicography*, 3(4), 1990.

[Palmer *et al.*, 1993] M. Palmer, R. Passonneau, C. Weir, and T. Finin. The kernel text understanding system. *Artificial Intelligence*, 63:17–68, 1993.

[Palmer, 1997] D. Palmer. A trainable rule-based algorithm for word segmentation. In *Proceedings of ACL/EACL97*, 1997.

[Perlmutter, 1983] D. Perlmutter. *Studies in Relational Grammar 1.* U. Chicago Press, 1983.

[Ramshaw and Marcus, 1995] L. Ramshaw and M. Marcus. Text chunking using transformation-based learning. In *Proc. of the 3rd Workshop on Very Large Corpora*, pages 82–94, Cambridge, MA, USA, 1995.

[Wolff *et al.*, 1995] S. Wolff, C. Macleod, and A. Meyers. *Comlex Word Classes.* C.S. Dept., New York U., Feb. 1995. prepared for the Linguistic Data Consortium, U. Pennsylvania.

[Yeh and Vilain, 1998] A. Yeh and M. Vilain. Some properties of preposition and subordinate conjunction attachments. In *COLING-ACL'98*, pages 1436–1442, Montréal, Canada, 1998.