Deep Semantic Interpretations Of Legal Texts

L. Thorne McCarty
Department of Computer Science
Rutgers University
New Brunswick, NJ 08903, USA
mccarty@cs.rutgers.edu

ABSTRACT
One of the main obstacles to progress in the field of artificial intelligence and law is the natural language barrier, but the technology of natural language processing has advanced recently. In this paper, we will show that a state-of-the-art statistical parser can handle even the complex syntactic constructions of an appellate court judge, and that a deep semantic interpretation of the full text of a judicial opinion can be computed automatically from the output of the parser. Our ultimate goal is to use this semantic interpretation to extract from a judicial opinion precisely the information that a lawyer wants to know about a case.

Keywords
Natural language processing, computational semantics, information extraction, legal texts.

1. INTRODUCTION
One of the main obstacles to progress in the field of artificial intelligence and law is the natural language barrier. Since the raw materials of the law are embodied in natural language — cases, statutes, regulations, etc. — the designer of a knowledge-based legal information system today must translate them, by hand, into a formal language, just to get started.

Several researchers have attempted to circumvent this problem by automating the translation process, or some part of it. In [10], for example, Daniels and Rissland used a combination of case-based reasoning (CBR) and information retrieval (IR) techniques to find relevant cases in a database and to focus attention on the most relevant passages in the text. In their work on the History Assistant project at West [11, 1], Jackson and Al-Kofahi, *et al.*, used partial parsing techniques to extract both "direct history" and "treatment history" from the full text of a case. In [5], Bruninghaus and Ashley used information extraction (IE) technology to identify factors in legal texts that could be used as input to a case-based reasoning system. Other relevant work includes: Moens, *et al.* [17]; van Engers, *et al.* [23]; Biagioli, *et al.* [2]; and Quaresma, *et al.* [19]. Notice, however, that none of these researchers have attempted to tackle the natural language processing (NLP) problem head on, presumably because they assumed that full-scale NLP was just too difficult in a domain as complex as the law.

This assumption is no longer valid, as the present paper will demonstrate. We will show that a state-of-the-art statistical parser [7, 9] can handle even the complex syntactic constructions of an appellate court judge, and that a semantic interpretation of the full text of a judicial opinion can be computed automatically from the output of the parser. Our ultimate goal is to use this semantic interpretation to extract from a judicial opinion precisely the information that a lawyer wants to know about a case.

Section 2 of the paper describes our initial legal corpus, and the preprocessing that is necessary before attempting a syntactic parse. Section 3 then discusses the parser and its accuracy, and illustrates its performance with a typical sentence from our corpus. The main technical contribution of the paper is in Section 4, in which we present our notion of a "deep semantic interpretation," and explain how it is computed. In particular, we discuss in detail in Section 4.1 the *quasi-logical forms* that are used to represent the semantics of a sentence, and we describe in Section 4.2 the *definite clause grammars* that are used to compute these quasi-logical forms.

Computing a deep semantic interpretation of a legal text is not an end in itself, of course. In Section 5 we describe, briefly, our plans to use these semantic interpretations to extract information from a case, which is the main thrust of our current research.

2. AN INITIAL LEGAL CORPUS
For reasons that will become apparent in Section 5 below, we are primarily interested in *federal civil* cases in the *appellate* courts in the United States. Our current data base thus consists of one month of judicial opinions from the United States Supreme Court, chosen (arbitrarily) from May, 1999; two months of judicial opinions from the United States Court of Appeals for the Second Circuit, chosen from May and June, 1999; and two months of judicial opinions from the United States Court of Appeals for the Third Circuit, chosen from

		cases	words	sentences	parse trees	null parses
SupCt	May 1999	9	35,796	1,213	1,141	5.94%
2dCirc	May 1999	31	66,333	2,341	2,170	7.30%
	June 1999	26	108,778	3,983	3,800	4.59%
3dCirc	May 1999	15	71,038	2,443	2,286	6.43%
	June 1999	30	148,320	5,382	5,132	4.65%
		111	430,265	15,362	14,529	5.42%

Table 1: Statistics on the current legal corpus.

May and June, 1999. We have excluded all criminal cases decided during these time periods, and we have excluded all cases with short *per curiam* opinions, or with no opinions at all. Thus every text in our database includes at least one substantial judicial opinion that describes the procedural history of the case, the claims and contentions of the parties, and the law and the facts that were in dispute. With these inclusions and exclusions, our current corpus consists of 111 cases. (Thanks to Stephen Max for his work on assembling and processing the cases.)

We have also started to assemble a statutory corpus. So far this includes the United States Copyright Act, and the Uniform Commercial Code, Article 3 (Negotiable Instruments) and Article 9 (Secured Transactions). We have only run a few small experiments with the parser on the statutory materials, but we plan to do more in the future. (Thanks to Samir Patel for his work on assembling and processing the statutes.)

The cases require a certain amount of preprocessing before we can use them. For example, judicial opinions are littered with case citations, sprinkled randomly through the text, and we have spliced these out to produce sentences with a more normal syntactic structure. Preprocessing also includes the determination of sentence boundaries [22], and an initial run through a part-of-speech tagger [21]. This preprocessing has been partially automated, but some manual editing is still necessary. For example, since federal judges tend to write verbose prose, some of their sentences exceed a length limit of 100 words imposed by the parser. Although we could certainly increase this limit, to do so would have a negative impact on processing time, and thus we are currently splitting long sentences into shorter ones, by hand, before submitting the cases to the parser. Finally, because of another threshold set in the parser, a sentence with a very low parse probability simply produces a null parse. Our current policy is to leave these sentences alone, since it is usually not clear how to produce a more reliable output in such a case.

Table 1 shows the current status of our corpus. The table lists those cases that have been successfully parsed, in each jurisdiction, and shows the total number of words and sentences in each, the number of nonempty parse trees produced by the parser, and the percentage of sentences for which the parse is null. Although we have now successfully parsed 100% of the cases in our database, the manual effort is excessive, and we are actively looking for ways to automate more of the preprocessing phase.

3. SYNTACTIC ANALYSIS

The main problem in parsing a sentence in natural language is the ambiguity that is inherent in any grammar that is rich enough to provide broad coverage of unrestricted text. The best current approach to this problem is *statistical*: a statistical model is constructed that assigns a probability to each possible parse tree; the parameters of the model are estimated from a tagged corpus, such as the Penn Wall Street Journal Treebank [12]; and the most probable parse tree, according to this model, is selected as the most plausible parse. For this approach to work, however, the statistical model must be carefully constructed, and it must be firmly grounded in linguistic theory. One of the best examples in the current literature is the *lexicalised head-driven statistical model* developed by Michael Collins in his dissertation in 1999 [7, 9].

It is Collins' statistical parser that we have used for the syntactic analysis of our legal corpus.[1] More specifically, we have used *Model 3* of Collins' parser, which handles the *complement/adjunct* distinction as well as the phenomenon of *wh-movement*. For an example of the parser's performance, here is a typical sentence from a case in our corpus, *Cleveland v. Policy Management Systems Corp.*, 526 U.S. 795 (1999):

> She has also brought this ADA suit in which she claims that her former employer, Policy Management Systems Corporation, discriminated against her on account of her disability.

The output is shown in Figure 1. The labels here, such as `NP`, `VP`, `ADVP`, `SBAR`, `PP`, etc., are derived from the Penn Treebank, and the numerical values on the nodes of the tree are log-probabilities. The suffix `-A` on some of the labels denotes a complement, and the pairs of integers on nonterminal nodes, such as `~3~1`, `~2~2`, etc., specify the number of subtrees attached at that node and the position of the *head* subtree among them. (There are actually two trees in this Figure, and it is necessary to merge them to extract all of the information available in the output of the parser. Since our subsequent processing of the parser's output is coded in Prolog, however, we immediately convert these two trees into a single Prolog term, taking care of the merger as we do so.)

[1]The source code for Collins' parser is available at ftp://ftp.cis.upenn.edu/pub/mcollins/PARSER.tar.gz. An alternative is to work with a re-implementation of Collins' parser by Daniel Bikel [3].

```
PROB 24150 -168.707 0
TOP -168.707 S -165.479 NP-A -0.020374 NPB -9.7834e-05 PRP 0 She
      VP -159.147 VBZ 0 has
         ADVP -0.000725472 RB 0 also
         VP-A -144.414 VBD 0 brought
              NP-A -140.331 NPB -8.83481 DT 0 this
                       NN 0 ADA
                       NN 0 suit
                   SBAR -122.939 WHPP -0.0292426 IN 0 in
                             WHNP -0.0152275 WDT 0 which
                        S-A -113.88 NP-A -0.0133551 NPB -0.00262232 PRP 0 she
                            VP -108.127 VBZ 0 claims
                               SBAR-A -106.51 IN 0 that
                                      S-A -98.0122 NP-A -45.2292 NPB -10.3173 PRP$ 0 her
                                                   JJ 0 former
                                                   NN 0 employer
                                               NP -27.2803 NPB -27.0538 NNP 0 Policy
                                                      NNP 0 Management
                                                      NNPS 0 Systems
                                                      NNP 0 Corporation
                                          VP -39.4834 VBN 0 discriminated
                                             PP -5.98807 IN 0 against
                                                NP-A -0.0307972 NPB -0.000116521 PRP 0 her
                                             PP -22.3409 IN 0 on
                                                NP-A -15.4045 NPB -1.89918 NN 0 account
                                                     PP -10.9464 IN 0 of
                                                        NP-A -7.48789 NPB -6.84328 PRP$ 0 her
                                                                 NN 0 disability
```

```
(TOP~has~1~1 (S~has~2~2 (NP-A~She~1~1 (NPB~She~1~1 She/PRP ) ) (VP~has~3~1 has/VBZ (ADVP~also~1~1
also/RB ) (VP-A~brought~2~1 brought/VBD (NP-A~suit~2~1 (NPB~suit~3~3 this/DT ADA/NN suit/NN )
(SBAR~in~2~1 (WHPP~in~2~1 in/IN (WHNP~which~1~1 which/WDT ) ) (S-A~claims~2~2 (NP-A~she~1~1
(NPB~she~1~1 she/PRP ) ) (VP~claims~2~1 claims/VBZ (SBAR-A~that~2~1 that/IN (S-A~discriminated~2~2
(NP-A~employer~2~1 (NPB~employer~3~3 her/PRP$ former/JJ employer/NN ,/PUNC, ) (NP~Corporation~1~1
(NPB~Corporation~4~4 Policy/NNP Management/NNP Systems/NNPS Corporation/NNP ,/PUNC, ) ) )
(VP~discriminated~3~1 discriminated/VBN (PP~against~2~1 against/IN (NP-A~her~1~1 (NPB~her~1~1 her/PRP
) ) ) (PP~on~2~1 on/IN (NP-A~account~2~1 (NPB~account~1~1 account/NN ) (PP~of~2~1 of/IN (NP-
A~disability~1~1 (NPB~disability~2~2 her/PRP$ disability/NN ./PUNC. ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) )
```

Figure 1: Syntactic analysis of a sentence from 526 U.S. 795 (1999).

How accurate is Collins' parser on sentences from judicial opinions? Since we do not have an annotated test set of judicial opinions, we cannot answer this question quantitatively. In Collins' own work, the accuracy of his parser was evaluated by training it on Sections 02-21 of the Wall Street Journal corpus (approximately 40,000 sentences) and then testing it on Section 23 (2,416 sentences). His overall results for *labeled recall* and *labeled precision* were 88.0% and 88.3%, respectively. However, the parser performed much better on the core structure of a sentence than it did on the fringes. The recall/precision for verb complements was 93.76/92.96, and for other complements it was 94.74/94.12. But for prepositional phrase attachments the recall/precision was 82.29/81.51, and for verb adjuncts other than prepositional phrases it was 75.11/78.44. The worst performance on these measures was for coordinated conjunction, where the recall/precision was only 61.47/62.20.

A qualitative analysis of our own results is consistent with these figures: Applied to judicial opinions, the parser is very good on the internal structure of sentences, but it is weaker on prepositional phrase attachments and coordinated conjunctions. Of course, we would not expect the parser to do quite as well as 88.0% recall and 88.3% precision on sentences from legal texts, since this is not the genre of the training set, but we conjecture that there is enough "legal" terminology in Wall Street Journal news stories that most of the necessary statistical information is available there.

In any event, there are several steps that could be taken to improve the accuracy of the parser on legal texts:

1. We could add a weighted sample of annotated legal sentences to the training set.
2. We could compute the "n-best" parse trees using Model

```
sterm(brought,A,
      [nterm(She,B,[])
       /det(null,prp),
       nterm(ADA,C,[]) &
       nterm(suit,D,[])
       && E^F^pterm(in,F,
                   [E,
                    nterm(which,D,[])]) &
           sterm(claims,E,
                 [nterm(she,G,[])
                  /det(null,prp),
                  sterm(discriminated,H,
                       [aterm(former,I,[J]) &
                        nterm(employer,J,[])
                        && nterm(Policy,K,[]) &
                           nterm(Management,L,[]) &
                           nterm(Systems,M,[]) &
                           nterm(Corporation,N,[])
                           /det(null,nnp)
                        /det(her,nn)])
                  && O^pterm(against,O,
                            [H,
                             nterm(her,P,[])
                             /det(null,prp)])
                  && Q^pterm(on,Q,
                            [H,
                             nterm(account,R,[])
                             && S^pterm(of,S,
                                        [R,
                                         nterm(disability,T,[])
                                         /det(her,nn)])
                             /det(null,nn)])
                  /[]])
            /[present]
       /det(this,nn)])
&& aterm(also,U,[A])
/[present,perfect]
```

Figure 2: Semantic interpretation of a sentence from 526 U.S. 795 (1999).

3 of Collins' parser, and then "rerank" the output by applying a discriminant function trained on legal texts. See [8] and [6] for a discussion of the improvements in recall and precision that are possible using this technique.

3. We could carry the "n-best" parse trees forward, with their probabilities attached, and use these as input into the subsequent stages of semantic and pragmatic processing.

We plan to investigate these various alternatives in our ongoing research.

4. SEMANTIC INTERPRETATION

Our own work begins where the work on the statistical parser leaves off. Figure 2 shows a semantic interpretation that was computed by our system using as input the parse tree in Figure 1. The data structure here is usually referred to as a *quasi-logical form* or *QLF*. Usually, a quasi-logical form is intended as an intermediate step on the way to a full *logical form*, which would include quantifiers, modalities, and many other features that are needed in a comprehensive knowledge representation language. (See [13].) However, our current intention is to use these semantic interpretations as inputs to an information extraction system (see Section 5), and thus our requirements are slightly different. The important questions are: (i) does the quasi-logical form capture in a convenient format the semantic information that will be needed by the information extraction system? and (ii) can the quasi-logical form be computed easily from the output of the statistical parser?

In Section 4.1, we will describe our quasi-logical forms in detail, as a step towards answering the first question, and in Section 4.2 we will answer the second question.

4.1 Quasi-Logical Forms

The quasi-logical form in Figure 2 is based loosely on my *Language for Legal Discourse (LLD)*, [13, 14, 16], which was originally developed as a knowledge representation language for legal domains. The basic construct is a *term*, of which there are four types: *sterm*, *nterm*, *aterm* and *pterm*. Each term contains three components: a *lexical item*, a *variable* and a *list of constituents*. The *sterm*, which is derived from a syntactic sentence, represents a relation: the lexical item is the name of the relation, and the list of constituents is a list of its arguments. The variable in this case represents a *relationship*, i.e., a particular instance of the relation. The *nterm*, which is derived from a syntactic noun, represents an object: the lexical item is the name of the object, and the variable represents a particular instance of the object. In the example in Figure 2, all the lists of object constituents are empty, but in other cases they might not be. For example, a "date" object could have constituents for days, months and years. Also, an event such as "issue" could be converted into an object called "issuance", and a relation such as "own" could be converted into an object called "ownership", in which case the arguments of the relation would be converted into constituents of the object. (Hence the slogan: "All relations are objects; all objects are relations.") The *aterm* represents a modifier, such as an adjective or an adverb. Its sole constituent is the object that it modifies. For example, in Figure 2, the *aterm* "former" modifies the object denoted by the variable `J`, which is an instance of the *nterm* "employer", and the *aterm* "also" modifies the object denoted by the variable `A`, which is an instance of the *sterm* "brought". Note that an *aterm* also contains a variable, which can be viewed as denoting an instance of the property that is being used as a modifier, e.g., an instance of the property "former", or an instance of the property "also". (Thus adverbs can modify adjectives.) Finally, a *pterm* represents the class of binary relations that correspond to prepositional phrases. These, too, have constituents, and they have instance variables, which represent relationships, and which can, in principle, be modified by other terms. For example, the variable `O` in Figure 2 denotes an instance of the binary relation "against".

Given these basic building blocks, a complete quasi-logical form is assembled by means of two operations: (1) an expression can be embedded in the list of constituents of a term; and (2) an expression can be adjoined, externally, either before or after a term. (The distinction between these two operations corresponds to the linguistic distinction between *complements* and *adjuncts*.) The expressions adjoined before the term, which are called *premodifiers*, are linked by a right associative operator: `&`. The expressions adjoined after the term, which are called *postmodifiers*, are linked by a left associative operator: `&&`. In Figure 2, these sequences of expressions are pretty-printed without using parentheses, but you can always locate the *base* of the sequence by following a chain of `&`s until you get to the last term before the beginning of a chain of `&&`s. In addition, when a *pterm* or an *sterm* is the base term in an expression that includes a sequence of modifiers, either *pre* or *post* and possibly null, that expression has prepended to it the instance variable of its base *pterm* or *sterm*. For example, in Figure 2, the variable `F` for "in" is prepended in the form `F^`, and the variable `E` for "claims" is prepended in the form `E^`. This notation is intended to suggest a *lambda expression*. The intuitive idea is that we might want to move these expressions around and attach them somewhere else, and the lambda variable gives us an operational handle on the base term that is independent of the surrounding sequence of modifiers.

There are two additional components of the quasi-logical form that need to be explained: First, since we are not attempting to analyze the quantificational structure of a sentence, we have opted simply to store all information about determiners as syntactic features attached to *nterm* expressions. For example, in Figure 2, the feature `det(this,nn)` attached to the expression for "ADA suit" means that the determiner for this expression is "this" and the syntactic category of the head noun is *singular common*. We have also extended this mechanism to possessive personal pronouns: Note the feature `det(her,nn)` attached to both the expression for "former employer" and the expression for "disability". Second, we have stored all information about verb tenses as syntactic features attached to *sterm* expressions. For example, in Figure 2, the feature `[present,perfect]` records the tense of the verb phrase "has brought". These tenses are computed from a grammar of the English auxiliary verb system, based on [20]. (Notice that there is an apparent error here in the tense of the verb "discriminated", which should be recorded as `[past]` but is instead recorded as `[]`. This is because the word "discriminated" was tagged as 'VBN' instead of 'VBD' in the syntactic parse tree. See Figure 1.)

The variables are the key to understanding what a quasi-logical form means. If it is possible, with reasonable certainty, to identify references solely from the syntactic analysis of a sentence, then the variables in the quasi-logical form are unified. Otherwise, the variables are kept distinct. For example, in Figure 2, the prepositional phrases containing the lexical items "against" and "on" are assumed to refer to the *sterm* "discriminated", and thus they share the variable `H` with the *sterm* variable `H`. But there is no way to identify references in the phrase "Policy Management Systems Corporation", and thus the variables `K`, `L`, `M` and `N` are kept distinct. A slightly more complicated example appears in the *wh*-clause in Figure 2. The *nterm* "which" shares the variable `D` with the *nterm* "suit", and the variable `E` in the *sterm* "claims" also appears as the first constituent of the *pterm* "in". Taken together, this means that the "claims" relation `E` is "in" the "suit" object `D`. Of course, variables that are distinct in a quasi-logical form do not necessarily refer to distinct objects in the world, since we can easily imagine subsequent operations that might identify some of these references. For example, the variable `B` in the *nterm* "She" might subsequently be unified with the variable `G` in the *nterm* "she". A quasi-logical form is therefore a pragmatic semantic compromise. The point is to make minimal commitments, with reasonable certainty, while leaving the more uncertain alternatives open.

Two additional examples are shown in Figure 3. These quasi-logical forms represent the following two sentences from *Monterey v. Del Monte Dunes at Monterey, Ltd.*, 526 U.S. 687:

> The petitioner contends that the regulatory takings claim should not have been decided by the jury and that the Court of Appeals adopted an erroneous standard for regulatory takings liability.

```
sterm(contends,A,
      [nterm(petitioner,B,[])
       /det(The,nn),
       sterm(decided,C,
             [D,
              aterm(regulatory,E,[F]) &
              nterm(takings,G,[]) &
              nterm(claim,F,[])
              /det(the,nn)])
       && H^pterm(by,H,
                  [C,
                   nterm(jury,I,[])
                   /det(the,nn)])
       /[modal(should),negative,perfect,passive]
       AND
       sterm(adopted,J,
             [nterm(Court,K,[])
              && L^pterm(of,L,
                         [K,
                          nterm(Appeals,M,[])
                          /det(null,nnps)])
              /det(the,nnp),
              aterm(erroneous,N,[O]) &
              nterm(standard,O,[])
              && P^pterm(for,P,
                         [O,
                          aterm(regulatory,Q,[R]) &
                          nterm(takings,S,[]) &
                          nterm(liability,R,[])
                          /det(null,nn)])
              /det(an,nn)])
       /[past]])
/[present]
```

```
sterm(ruled,A,
      [nterm(court,B,[])
       /det(The,nn),
       sterm(presented,C,
             [D,
              aterm(sufficient,E,[F]) &
              nterm(evidence,F,[])
              /det(null,nn)])
       && G^pterm(to,G,
                  [C,
                   nterm(jury,H,[])
                   /det(the,nn)])
       && I^J^pterm(from,J,
                    [I,
                     nterm(which,C,[])]) &
            aterm(reasonably,K,[I]) &
            sterm(decided,I,
                  [nterm(it,L,[])
                   /det(null,prp),
                   nterm(each,M,[])
                   && N^pterm(of,N,
                              [M,
                               nterm(questions,O,[])
                               && P^pterm(in,P,
                                          [O,
                                           Q^nterm(Del,R,[]) &
                                             nterm(Monte,S,[]) &
                                             nterm(Dunes,T,[]) &
                                             pterm('POS',Q,
                                                   [U,
                                                    V])
                                           /det(null,pos) &
                                           nterm(favor,V,[])
                                           /det(null,nn)])
                               /det(these,nns)])
                   /det(null,dt)])
          /[modal(could),perfect]
       /[past,perfect,passive]])
/[past]
```

Figure 3: Semantic interpretation of two sentences from 526 U.S. 687 (1999).

> The court ruled that sufficient evidence had been presented to the jury from which it reasonably could have decided each of these questions in Del Monte Dunes' favor.

These examples illustrate several additional features of our semantic interpretation. The first example shows how the conjunction "and" in the original sentence is translated into a quasi-logical "AND" that connects *sterm* expressions. A similar operation applies to *nterm* expressions, etc. Note also how the complex verb phrase "should not have been decided" is translated into the feature

```
[modal(should),negative,perfect,passive]
```

attached to the *sterm* "decided". The second example illustrates a special construction for the possessive phrase: "Del Monte Dunes' favor". The base term in the possessive construction is a *pterm* with a reserved keyword `'POS'` as its lexical item, asserting that some variable `U` is in "possession" of the variable `V`, which stands for the *nterm* "favor". The variable `U` is related, somehow, to the variables `R`, `S` and `T`, but as in the case of all noun-noun modifier expressions, it is impossible to determine these relationships by a purely syntactic analysis. Thus, once again, the quasi-logical form is making a minimal commitment, with reasonable certainty, while leaving the more uncertain alternatives open.

There is an interesting mistake here in the attachment of the prepositional phrase "in Del Monte Dunes' favor". The variable `O` in the *pterm* refers to the *nterm* "questions", but it is not the "questions" that are "in Del Monte Dunes' favor". Most likely, the prepositional phrase should have been attached as a postmodifier to the *sterm* "decided". Although it is possible that an improved syntactic analysis would correct this error, we cannot rely on syntax alone to attach prepositional phrases with complete accuracy. One alternative is to work directly with the quasi-logical form to explore additional attachments. In this case, if we replaced the variable `O` in the *pterm* with the variable `M` or the variable `I`, we would effectively be moving the prepositional phrase to a higher node in the parse tree. This option could be made available as part of the pragmatic processing that is intended to follow the construction of a semantic interpretation.

4.2 Definite Clause Grammars

How are the quasi-logical forms in our system computed? The basic technique is well known in the literature on *definite clause grammars* or *DCG*s [18]. Each node in the parse tree is assigned a semantic interpretation, and the interpretation of a parent node is defined compositionally in terms of the interpretations of its children. Moreover, since a definite clause grammar is just a Prolog program, the semantic step from child node to parent node can be effectuated by *unification*.

For a simple example, here is the rule that defines the interpretation of a sentence given the interpretations of its *subject complement* noun phrase and its *head* verb phrase:

```
cat 'S'(Tense) :: E^Term *->
    comp 'NP'(Det) :: _^NP,
    head 'VP'(_Type,Tense) :: (NP/Det)^E^Term.
```

To understand this rule, read the symbol '`::`' as an operator that assigns the semantic structure on the right to the syntactic structure on the left, and read '`*->`' as the rewrite symbol of the grammar. Assume that the semantics of a noun phrase always takes the form `X^NP`, where `X` is the variable in the *nterm* `NP`. Assume that the semantics of a verb phrase always takes the form `Subj^E^Term`, where `Subj` is the first constituent in the *sterm* `Term`. Then the rule above says that the semantics of the sentence takes the form `E^Term`, with the following additional consequences: (i) the variable `X` is dropped; (ii) the feature `Det`, which has been propagated as part of the syntax of a noun phrase, now becomes part of the semantics of the verb phrase; and (iii) the first constituent in the *sterm* `Term` is instantiated to the semantic subject `NP/Det`.

For a more complex example, here are the three rules that are responsible for the interpretation of the *wh*-clause in Figure 2:

```
cat 'SBAR'(whpp) :: W^(E^(P^Term & S)/Tense) *->
    head 'WHPP' :: W^E^P^Term,
    comp 'S'(Tense) :: E^S.

cat 'WHPP' :: W^Subj^P^Term *->
    head 'IN'(_Word) :: Obj^Subj^P^Term,
    args 'WHNP' :: W^Obj.

cat 'WHNP' :: W^Term *->
    head 'WDT'(_Word) :: W^Term.
```

As an exercise, the reader should try to figure out how these three rules work, by comparing the parse tree in Figure 1 with the semantic interpretation in Figure 2.

The only difficulty in writing these rules is that they are quite numerous. We wrote the semantic rules by hand, but with the help of some automated tools. First, we took a large portion of our parsed legal corpus and tallied all the grammar rules that appeared there, in three variants: (i) main grammar rules, such as those shown above, with distinct heads and complements; (ii) modifier rules, with a base category and a nonterminal premodifier; (iii) modifier rules, with a base category and a nonterminal postmodifier. This was a large set, but not too large to handle, and the terminal modifiers, which are not included in these three variants, could be treated separately and (almost) uniformly. We also developed an extended language for the rule interpreter, so that we could write more expressive grammars than those shown above, if necessary. (This was how we wrote the rules to parse the English auxiliary verb system.) Finally, to assist in the writing of particular semantic constructs, we developed tools that could efficiently retrieve any fragment of any parse tree, anywhere in the corpus, if it satisfied the properties that we were interested in.

Our current *DCG* for semantic interpretation consists of approximately 700 rules. Of course, since this grammar was developed on a specific set of cases, it could always turn out to be missing a rule when applied to a new case. But this would not be a disaster, since the rule interpreter would simply insert a variable term into the quasi-logical form in such a situation. Also, since the rule interpreter stores the missing rule in an error log, we can subsequently add it to the grammar. As a result, the definite clause grammar gets better all the time, and when it fails, it fails "softly" by leaving a small gap in the quasi-logical form. Since the distribution of the grammar rules follows Zipf's law [24], these gaps are of minor importance. They are rare events, and they are often simply syntactic errors. The main structure of the quasi-logical form remains fairly robust.

The work on semantic interpretation most similar in spirit is probably [4], although it is difficult to make a direct comparison. Bos *et al.* use a Combinatory Categorial Grammar (CCG) for their parser, which is trained on a translation of the Penn Treebank, called the CCGbank, and produces a parse tree in the form of a CCG derivation. For their semantic interpretation, they attach λ-expressions to the leaf nodes of the parse tree and reformulate the combinatory rules at the interior nodes as functional applications. Thus the composition of semantic forms in their system uses the full machinery of the lambda calculus (α-conversion, β-conversion, etc.), whereas our system uses unification.

5. FUTURE WORK

We described the ultimate goal of our research in an earlier paper [15]. The challenge is to produce a *structured casenote*, which is a computational version of the traditional "brief" that first-year law students are taught to write as a summary of each case in their casebooks. Here is a short description from [15]:

> The traditional case brief focuses on the procedural context first: Who is suing whom, and for what? What is the plaintiff's legal theory? What facts does the plaintif allege to support this

theory? How does the defendant respond? How does the trial court dispose of the case? What is the basis of the appeal? What issues of law are presented to the appellate court? How does the appellate court resolve these issues, and with what justification?

Our initial legal corpus was chosen with this goal in mind, and our "deep semantic interpretations" of legal texts are intended to provide the information that is needed to answer these questions. Note that the semantic interpretation in Figure 2 includes a partial answer to the first two questions in the preceding list, and the semantic interpretations in Figure 3 include partial answers to some of the subsequent questions.

Thus the remaining challenge is: How can we extract the answers to these critical questions, automatically, from the semantic interpretations described in the present paper? This is the subject of our current and future research.

6. REFERENCES

[1] K. Al-Kofahi, B. Grom, and P. Jackson. Anaphora resolution in the extraction of treatment history language from court opinions by partial parsing. In *Proceedings of the 7th International Conference on Artificial Intelligence and Law (ICAIL '99)*, pages 138–146. ACM Press, 1999.

[2] C. Biagioli, E. Francesconi, A. Passerini, S. Montemagni, and C. Soria. Automatic semantics extraction in law documents. In *Proceedings of the 10th International Conference on Artificial Intelligence and Law (ICAIL '05)*, pages 133–140. ACM Press, 2005.

[3] D. M. Bikel. Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4):479–511, 2004.

[4] J. Bos, S. Clark, M. Steedman, J. R. Curran, and J. Hockenmaier. Wide-coverage semantic representations from a CCG parser. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*, pages 42–51, 2004.

[5] S. Bruninghaus and K. D. Ashley. Improving the representation of legal case texts with information extraction methods. In *Proceedings of the 8th International Conference on Artificial Intelligence and Law (ICAIL '01)*, pages 42–51. ACM Press, 2001.

[6] E. Charniak and M. Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180. Association for Computational Linguistics, June 2005.

[7] M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.

[8] M. Collins. Discriminative reranking for natural language parsing. In *Proceedings of the 17th International Conference on Machine Learning*, pages 175–182. Morgan Kaufmann, 2000.

[9] M. Collins. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637, 2003.

[10] J. J. Daniels and E. L. Rissland. Finding legally relevant passages in case opinions. In *Proceedings of the 6th International Conference on Artificial Intelligence and Law (ICAIL '97)*, pages 39–46. ACM Press, 1997.

[11] P. Jackson, K. Al-Kofahi, C. Kreilick, and B. Grom. Information extraction from case law and retrieval of prior cases by partial parsing and query generation. In *Proceedings of the 7th International Conference on Information and Knowledge Management (CIKM '98)*, pages 60–67. ACM Press, 1998.

[12] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[13] L. T. McCarty. A language for legal discourse I. Basic features. In *Proceedings of the 2nd International Conference on Artificial Intelligence and Law (ICAIL '89)*, pages 180–189. ACM Press, 1989.

[14] L. T. McCarty. An implementation of Eisner v. Macomber. In *Proceedings of the 5th International Conference on Artificial Intelligence and Law (ICAIL '95)*, pages 276–286. ACM Press, 1995.

[15] L. T. McCarty. L'indexation de la jurisprudence. In D. Bourcier, P. Hassett, and C. Roquilly, editors, *Droit et Intelligence Artificielle: Une Revolution de la Connaisance Juridique*, pages 191–200. Romillat, Paris, 2000. English version available at http://www.cs.rutgers.edu/pub/mccarty/nice98.pdf.

[16] L. T. McCarty. Ownership: A case study in the representation of legal concepts. *Artificial Intelligence and Law*, 10(1-3):135–161, 2002.

[17] M.-F. Moens, C. Uyttendaele, and J. Dumortier. Abstracting of legal cases: The SALOMON experience. In *Proceedings of the 6th International Conference on Artificial Intelligence and Law (ICAIL '97)*, pages 114–122. ACM Press, 1997.

[18] F. C. N. Pereira and S. M. Shieber. *PROLOG and Natural Language Analysis*. CSLI Publications, Stanford, CA, USA, 1987.

[19] P. Quaresma and I. P. Rodrigues. A question-answering system for Portuguese juridical documents. In *Proceedings of the 10th International Conference on Artificial Intelligence and Law (ICAIL '05)*, pages 256–257. ACM Press, 2005.

[20] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A Comprehensive Grammar of the English Language*. Longman, London, UK, 1985.

[21] A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In E. Brill and K. Church, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142. Association for Computational Linguistics, 1996.

[22] J. C. Reynar and A. Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 16–19. Morgan Kaufmann, 1997.

[23] T. M. van Engers, R. Gerrits, M. Boekenoogen, E. Glassée, and P. Kordelaar. POWER: Using UML/OCL for modelling legislation - an application

report. In *Proceedings of the 8th International Conference on Artificial Intelligence and Law (ICAIL '01)*, pages 157–167. ACM Press, 2001.

[24] G. K. Zipf. *Selected Studies of the Principle of Relative Frequencies of Language*. Harvard University Press, 1932.