Using Grammatical Relations to Compare Parsers

Judita Preiss*
Computer Laboratory
University of Cambridge
Judita.Preiss@cl.cam.ac.uk

Abstract

We use the grammatical relations (GRs) described in Carroll et al. (1998) to compare a number of parsing algorithms. A first ranking of the parsers is provided by comparing the extracted GRs to a gold standard GR annotation of 500 Susanne sentences: this required an implementation of GR extraction software for Penn Treebank style parsers. In addition, we perform an experiment using the extracted GRs as input to the Lappin and Leass (1994) anaphora resolution algorithm. This produces a second ranking of the parsers, and we investigate the number of errors that are caused by the incorrect GRs.

1 Introduction

We investigate the usefulness of a grammatical relation (GR) evaluation method by using it to compare the performance of four full parsers and a GR finder based on a shallow parser.

It is usually difficult to compare performance of different style parsers, as the output trees can vary in structure. In this paper, we use GRs to provide a common basis for comparing full and shallow parsers, and Penn Treebank and Susanne structures. To carry out this comparison, we implemented a GR extraction mechanism for Penn Treebank parses. Evaluating parsers using GRs as opposed to crossing brackets or labelled precision/recall metrics can be argued to give a more robust measure of performance (Carroll et al., 1998), (Clark and Hockenmaier, 2002). The main novelty of this paper is the use of the Carroll et al's GR evaluation method to compare the Collins model 1 and model 2, and Charniak parsers.

An initial evaluation is provided by comparing the extracted GRs to a gold standard GR annotation of 500 Susanne sentences due to Carroll et al. To gain insight into the strengths and weaknesses of the different parsers, we present a breakdown of the results for each type of GR. It is not clear whether the ranking produced from the gold standard evaluation is representative: there may be corpus effects for parsers not trained on Susanne, and real life applications may not reflect this ranking. We therefore perform an experiment using the extracted GRs as input to the Lappin and Leass (1994) anaphora resolution algorithm. This produces a second ranking of the parsers, and we investigate the number of errors that are caused by incorrect GRs.

We describe the parsers and the GR finder in Section 2. We introduce GRs in Section 3 and briefly describe our GR extraction software for Penn Treebank style parses. The evaluation, including a description of the evaluation corpus and performance results, is presented in Section 4. The results are analyzed in Section 5 and a performance comparison in the context of anaphora resolution is presented in Section 6. We draw our conclusions in Section 7.

*This work was supported by UK EPSRC project GR/N36462/93 'Robust Accurate Statistical Parsing'.

Parser	Corpus	LR	LP	CB	0CB	2CB
sentences ≤ 40 words						
CH	WSJ	90.1	90.1	0.74	70.1	89.6
C1	WSJ	87.52	87.92	0.96	64.86	86.19
C2	WSJ	88.07	88.35	0.95	65.84	86.64
sentences ≤ 100 words						
CH	WSJ	89.6	89.5	0.88	67.6	87.7
C1	WSJ	87.01	87.41	1.11	62.17	83.86
C2	WSJ	87.60	87.89	1.09	63.20	84.60
BC evaluation						
BC	Susanne	74.0	73.0	1.03	59.6	–

Table 1: Summary of Published Results (LR = labelled recall, LP = labelled precision, CB = crossing brackets)

	BC[a]	CH[b]	C1 & C2[c]	BU
Grammar	Unification-based, PoS and punct labels	Generative, 3rd order	Generative, 0th order	N/A
Algorithm	LR parser	Chart parser		Shallow parser
Tagger	Acquilex (CLAWS-II) (Elworthy, 1994)	own	Ratnaparkhi (1996).[d]	Memory-based (Daelemans et al., 1996)
Training	Susanne (Sampson, 1995)[e]	Sections 2–21 of the Wall Street Journal portion of the Penn Treebank (Marcus et al., 1993)		Sections 10–19 of the WSJ corpus of the Penn Treebank II

[a]Available from `http://www.cogs.susx.ac.uk/lab/nlp/rasp/`
[b]Available from `ftp://ftp.cs.brown.edu/pub/nlparser/`
[c]Available from `ftp://ftp.cis.upenn.edu/pub/mcollins/misc/`
[d]Available from `http://www.cis.upenn.edu/~adwait/`
[e]Note that the Briscoe and Carroll grammar is manually created, and Susanne was used for development.

Table 2: Parser Descriptions

2 Tools

In this work we compare four full parsers from which GRs are extracted by walking over the trees. These parsers are Briscoe and Carroll (1993) (BC), Charniak (2000) (CH), model 1 and model 2 of Collins (1997) (C1 and C2).[1] A summary of published performance results can be found in Table 1. We also include in our comparison a GR finder (Buchholz, 2002) (BU) based on a shallow parser (Daelemans, 1996), (Buchholz et al., 1999). Table 2 summarizes the grammar, the parsing algorithm, the tagger and the training corpus for all the parsers that we investigate.

[1]Note that Collins' model 1 and Collins' model 2 are considered as two different parsers.

3 Grammatical Relations

Lin (1995) proposed an evaluation based on grammatical dependencies, in which syntactic dependencies are described between heads and their dependents. This work was extended by Carroll et al. (1998), and it is this specification called grammatical relations which we employ in our work. An example, for the sentence *John gave Mary the book*, can be seen in Figure 1.

Both the Briscoe and Carroll parser and

Sentence: John gave Mary the book.

Grammatical relations:

```
(ncsubj gave John)
(dobj gave Mary)
(obj2 gave book)
(detmod book the)
```

Figure 1: Sample GR output

Buchholz's GR finder already output GRs in the desired format. Although Buchholz's work has focused mainly on extracting relations involving verbs, some non-verb relations (e.g. *detmod*) are also produced by the chunker she employs (Veenstra and van den Bosch, 2000).

Therefore, to carry out a GR comparison, we need to extract GRs from Penn Treebank style parses. We manually created rules which find the relevant heads and their dependants by traversing the parse tree (for example, the NP in a S $\rightarrow$ NP VP rule gives an instance of the *ncsubj* relation). In cases where a distinction is difficult/impossible to make from a Penn Treebank tree (e.g. *xcomp* vs *xmod*), we sacrificed recall for precision and only encoded rules which cause as few misclassifications as possible.[2]

Similar work has been carried out by Blaheta and Charniak (2000) who used statistical methods to add function tags to Penn Treebank I style parses; however, as well as converting the tags into a Carroll et al. format, we would need to add extra rules to extract other GRs needed for our application described in Section 6. E.g. the direct object is not immediately apparent from Penn Treebank II tags.

We restrict the GRs we extract from the Penn Treebank to those that are necessary for the anaphora resolution application (the object relations, the complement relations and the *ncmod* relation), and those that are a simple by-product of extracting the necessary relations (e.g. *aux*).

[2]These kind of errors caused by the GR extraction rules may be responsible for degraded performance.

4 Evaluation

As part of the development of their parser, Carroll et al. have manually annotated 500 sentences with their GRs.[3] The sentences were selected at random from the Susanne corpus subject to the constraint that they are within the coverage of the Briscoe and Carroll parser.[4] We used our own evaluation software which only scores correct an exact match of the output with the gold standard. This has caused some differences in performance with previously published results, for example the Briscoe and Carroll GRs do not produce the expected conjunction in the *conj* relation, causing the system to score zero.

The results of all systems are presented in Table 3. For each system, we present two figures: precision (the number of instances of this GR the system correctly annotated divided by the number of instances labelled as this GR by the system), and recall (the number of instances of this GR the system correctly annotated divided by the number of instances of this GR in the corpus). In the #occs column of the table, we also present the number of occurrences of each GR in the 500 sentence corpus. A dash (–) indicates that a certain GR annotation was not present in the answer corpus at all.[5] We also show the mean μ precision and recall for each system, and the weighted mean μ_W, where precision and recall values are weighted by the number of occurrences of each GR.

To obtain a ranking of the parsers, we compare $F_{\beta=1}$ using the *t*-test. The 500 sentence corpus is split into 10 segments and an F-measure is computed for each algorithm on all segments. These are then compared using the *t*-test. The results are presented in Table 4, which is to be interpreted as follows:

[3]Available from `http://www.cogs.susx.ac.uk/lab/nlp/carroll/greval.html`

[4]The parser has a coverage of about 74% on Susanne.

[5]Note that we currently do not extract *cmod, conj, csubj, mod, subj, xmod* or *xsubj* from Penn Treebank parses. We have merged the *xcomp, ccomp* and *clausal* GRs to make the evaluation meaningful (no *clausal* tags appear in the gold standard).

GR	# occ	BC		BU		CH		C1		C2	
arg_mod	41	–	–	75.68	68.29	78.12	60.98	82.86	70.73	82.86	70.73
aux	381	87.06	84.78	93.70	89.76	89.86	83.73	87.00	86.09	89.89	86.35
clausal	403	43.27	52.61	75.79	71.46	62.19	43.67	50.57	32.75	49.11	27.30
cmod	209	38.28	23.45	55.71	18.66	–	–	–	–	–	–
conj	165	0.00	0.00	80.00	24.24	–	–	–	–	–	–
csubj	3	0.00	0.00	0.00	0.00	–	–	–	–	–	–
detmod	1124	91.15	89.77	92.41	90.93	90.19	87.54	92.09	89.06	92.15	88.79
dobj	409	85.83	78.48	88.42	76.53	84.43	75.55	86.16	74.57	84.85	75.31
iobj	158	31.93	67.09	57.75	51.90	27.53	67.09	27.09	69.62	27.01	70.25
mod	21	1.25	28.57	–	–	–	–	–	–	–	–
ncmod	2403	69.45	57.72	66.86	51.64	79.84	46.32	81.46	47.36	81.08	47.27
ncsubj	1038	81.99	82.47	85.83	72.93	81.80	70.13	79.19	65.99	81.29	69.46
obj2	19	27.45	73.68	46.15	31.58	61.54	42.11	81.82	47.37	61.54	42.11
subj	1	0.00	0.00	–	–	–	–	–	–	–	–
xmod	128	13.64	2.34	69.23	7.03	–	–	–	–	–	–
xsubj	5	–	–	50.00	40.00	–	–	–	–	–	–
μ	407	35.71	40.06	58.60	43.43	40.97	36.07	41.77	36.47	40.61	36.10
μ_W	–	69.99	65.86	77.30	64.06	73.86	57.90	73.67	57.42	73.81	57.62

Table 3: GR Precisions and Recalls

	BC	BU
C2	85	–

This example means that the Collins model 2 parser does not outperform the Buchholz GR finder, but it outperforms the Briscoe parser with a statistical significance of 85%. Table 4 shows that the Buchholz' GR finder, based on a shallow parser, outperforms all the other parsers. This is followed in order by Charniak's, Collins' model 2, Collins' model 1, and then Briscoe and Carroll's.

	BC	BU	CH	C1	C2
BC	–	–	–	–	–
BU	99.5	–	99.5	99.5	99.5
CH	85	–	–	75	55
C1	70	–	–	–	–
C2	85	–	–	80	–

Table 4: t-tests for F-measure

5 Error Analysis

We investigated the cases where groups of systems failed to annotate some GRs (missing GRs) and cases where groups of systems returned the same wrong relation (extra GRs). The results of this are presented in Tables 5 and 6. In Table 5, we present the percentage of wrong cases covered by a particular combination of systems (i.e. BC represents the proportion of extra relations which were only suggested by the Briscoe and Carroll parser, whereas BC BU CH C1 C2 represents those extras which were suggested by all parsers.)[6] We present individual percentages, the percentage covered by the related Collins parsers (C1 C2), the Penn Treebank parsers (CH C1 C2) and all systems. The GRs wrongly suggested by all systems could be used to identify errors in the gold standard, since these break down into:

- Extra *aux* relations, where this is not marked up in the gold standard (e.g. *...receive... approval... to be printed...* is missing the

[6]Note that we are generating a probability distribution of same extra GRs over all system combinations. For example, C1 C2 represents the extra cases suggested by precisely these systems and so is independent of the percentage covered by CH C1 C2.

	BC	BU	CH	C1	C2	C1 C2	CH C1 C2	BC BU CH C1 C2
arg_mod	0.00	35.71	21.43	0.00	0.00	7.14	7.14	0.00
aux	27.78	7.78	7.78	14.44	1.11	7.78	2.22	14.44
clausal	48.25	11.87	7.59	5.06	2.92	7.20	7.39	0.00
detmod	22.94	14.22	15.14	1.38	0.92	3.21	12.84	10.55
dobj	24.65	10.56	14.08	4.93	4.23	7.75	9.15	3.52
iobj	14.93	2.99	10.87	0.85	2.13	12.58	17.06	4.26
ncmod	31.51	32.63	6.45	0.23	0.53	4.28	6.83	2.03
ncsubj	25.20	14.43	13.01	7.52	3.05	10.37	7.52	3.46
obj2	66.67	13.73	5.88	1.96	3.92	0.00	0.00	0.00

Table 5: Percentage of Extras

	BC	BU	CH	C1	C2	C1 C2	CH C1 C2	BC BU CH C1 C2
arg_mod	53.66	0.00	0.00	0.00	0.00	0.00	0.00	19.51
aux	18.00	9.00	8.00	2.00	1.00	4.00	7.00	20.00
clausal	16.71	0.26	0.26	0.00	1.57	13.84	23.76	15.40
detmod	17.75	7.79	11.69	0.87	2.16	3.46	11.26	21.21
dobj	12.09	12.09	4.95	3.30	1.65	4.40	11.54	20.88
iobj	12.15	18.69	1.87	0.93	0.00	2.80	6.54	17.76
ncmod	2.99	13.26	3.10	0.11	0.28	1.75	9.31	29.80
ncsubj	7.24	11.07	3.02	5.03	0.40	3.42	17.51	17.91
obj2	0.00	0.00	6.67	0.00	0.00	0.00	0.00	13.33

Table 6: Percentage of Missing

`(aux printed be)` relation).

- Extra *ncmod* relations, due to wrong identification of the head by the algorithms or in the gold standard.
- Extra *iobj* relations, due to a misclassification of an *ncmod* relation.

Table 6 classifies the cases of missing GRs and could therefore be used to discover missing classes of GRs, as well as mistakes in the gold standard. The main sources of errors are:

- Missing *ncmod* relations where the modifier is temporal, e.g. `(ncmod say Friday)`.
- Missing *detmods*, due to certain words not being assigned a determiner tag by the taggers. Examples of such words are *many* and *several*. This error creates extra *ncmod* relations instead.

The table also shows that the *clausal* relation would benefit from improvement since the *clausal* relations is frequently omitted from all the Penn Treebank parsers. However, in the case of this relation, we have sacrificed recall for precision.

6 Anaphora Resolution

We investigate the effect of using different parsers in an anaphora resolution system. This will indicate the impact of a change in parser performance on a real task: although one parser may have a marginally higher precision than another on a particular evaluation corpus, it is not clear whether this will be reflected by the results of a system which makes use of this parser, and which may work on a different corpus.

6.1 Lappin and Leass

We choose to re-implement a non-probabilistic algorithm due to Lappin and Leass (1994), because this anaphora resolution algorithm can be encoded in terms of the GR information (Preiss and Briscoe, 2003). For each pronoun, this algorithm uses syntactic criteria to rule out noun phrases that cannot possibly corefer with it. An antecedent is then chosen according to a ranking based on salience weights.

For all pronouns, noun phrases are ruled out if they have incompatible agreement features. Pronouns are split into two classes, lexical (reflexives and reciprocals) and non-lexical anaphors. There are additional syntactic filters for both of the two types of anaphors.

Factor	Weight
Sentence recency	100
Subject emphasis	80
Existential emphasis	70
Accusative emphasis	50
Indirect object/oblique	40
Head noun emphasis	80
Non-adverbial emphasis	50

Table 7: Salience weights

Candidates which remain after filtering are ranked according to their salience. A salience value corresponding to a weighted sum of the relevant feature weights (summarized in Table 7) is computed. If we consider the sentence *John walks*, the salience of John will be:

$$\begin{aligned} \text{sal(John)} &= w_{\text{sent}} + w_{\text{subj}} + w_{\text{head}} + w_{\text{non-adv}} \\ &= 100 + 80 + 80 + 50 \\ &= 310 \end{aligned}$$

The weights are scaled by a factor of $\left(\frac{1}{2}\right)^s$ where s is the distance (number of sentences) of the candidate from the pronoun.

The candidate with the highest salience is proposed as the antecedent.

6.2 Using GR Information

The algorithm uses GR information at two points: initially, it is used to eliminate certain intrasentential candidates from the candidates list. For example, in the sentence *She likes her*, *she* and *her* cannot corefer, which is expressed by a shared head in the following GRs:

```
(ncsubj like she)
(dobj like her)
```

Secondly, GR information is used for obtaining salience values. In the above sentence, we would use the *ncsubj* relation to reward *she* for being a subject and the *dobj* relation to give *her* points for accusative emphasis.

The algorithm makes use of the object relations (*ncsubj*, *dobj*, *obj2*, *iobj*), the complement relations (*xcomp*, *ccomp*, and *clausal*), and the non-clausal modifier *ncmod* relation.

6.3 Evaluation

	Sents	Prons
1	754	134
2	785	116
3	318	153
4	268	135
5	271	135

Table 8: Corpus Information

For this experiment, we use an anaphorically resolved 2400 sentence initial segment of the BNC (Leech, 1992), which we split into five segments containing roughly equal numbers of pronouns. The number of sentences and pronouns in each of the five segments is presented in Table 8.

	BC	BU	CH	C1	C2
1	60.45	63.43	62.69	62.69	61.19
2	50.86	52.59	54.31	55.17	54.31
3	69.93	69.93	69.28	67.32	69.28
4	67.41	65.19	69.63	63.70	66.67
5	54.81	52.59	50.37	51.85	51.85
μ	60.69	60.75	61.26	60.15	60.66
σ^2	52.36	48.87	60.66	32.83	45.73

Table 9: Anaphora Results

The results of the Lappin and Leass anaphora resolution algorithm using each of the parsers are presented in Table 9.[7] The 'algorithms' are only evaluated on pronouns

[7]In this case, Briscoe means the Lappin and Leass algorithm using the GRs generated by the Briscoe and Carroll algorithm, etc.

	BC	BU	CH	C1	C2
BC	–	–	–	60	0
BU	0	–	–	70	0
CH	60	65	–	75	75
C1	–	–	–	–	–
C2	–	–	–	70	–

Table 10: t-tests for Anaphora Resolution Performance

where all systems suggested an answer, so only precision is reported.[8] The difference between the 'worst' and the 'best' systems' mean μ performance is about 1%. However, the variance σ^2 (a measure of robustness of a system) is lowest for Collins' model 1. We again investigate the significance of the performance results using a t-test on our five segments, and the results can be seen in Table 10. The ranking obtained in this case indicates very small differences in performance between the algorithms.

6.4 Error Analysis

In our error analysis, we found that in 40% ($\frac{153}{385}$) of the errors the anaphora resolution algorithm made a mistake with all the parsers. This suggests that for a large number of pronouns, the error is with the anaphora resolution algorithm and not with the parser employed. The breakdown of the number of systems that suggested each mistake for each pronoun can be seen in Table 11.[9]

# Systems	0	1	2	3	4	5
# Pronouns	288	71	59	54	48	153

Table 11: Number of Mistaken Systems

It is also interesting to see the number of different antecedents suggested by the anaphora resolution algorithm using the various parsers (Table 12). We can see that there is a tendency to choose the same (potentially wrong) antecedent, since there are no cases where all versions of the Lappin and Leass algorithm chose different antecedents (versus 153 times all systems chose the wrong antecedent). The number of times that only one antecedent exists in the suggested answers is strikingly high. However, this may be slightly misleading, as a chosen pronominal antecedent (e.g. in *Mary*... She_1... She_2, She_2 will resolve to She_1) counts as identical whether or not it refers to the same entity. In scoring the anaphora resolution, if She_1 was previously wrongly resolved, She_2 is also treated as an error. This choice of evaluation method may be having an impact on our overall accuracy.

# Antecedents	0	1	2	3	4	5
# Pronouns	–	436	203	30	4	0

Table 12: Number of Different Antecedents

7 Conclusion

We have presented two evaluations of information derived from full and shallow parsers. The first compares the results of certain GRs against a gold standard, and the second investigates the change in accuracy of an anaphora resolution system when the parser is varied.

When the systems' F-measures were compared, we found that Buchholz' GR finder outperformed the conventional full parsers. This is an interesting result, which shows that accurate GRs can be obtained without the expense of constructing a full parse. The ranking between the Penn Treebank parsers obtained from the GR evaluation reflects the ranking obtained from a direct parser comparison (from Table 1).

In the task-based evaluation, the performance gap between the anaphora resolution algorithm using the various parsers narrowed. This may be due to the anaphora resolution algorithm making use of only certain instances of GRs which are 'equally difficult' for all parsers to extract.

We expect the results of the anaphora res-

[8]Systems attempt all pronouns which they are given; pronouns were only removed if the correct antecedent was wrongly tagged. Only about 10 pronouns were removed in this way.

[9]In this table, 1 system means only one system chose the wrong antecedent, etc.

olution experiment to be typical of parser applications that make use of a large number of types of GRs. Future work is required to evaluate parsers on applications that make use of just a few types of GRs, for example selectional preference based word sense disambiguation.

Acknowledgements

I would like to thank Sabine Buchholz for providing me with the output from her system. My thanks also go to Ted Briscoe for his insight into parsers, Ann Copestake and Joe Hurd for reading previous drafts of this paper.

References

D. Blaheta and E. Charniak. 2000. Assigning function tags to parsed text. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 234–240.

E. J. Briscoe and J. Carroll. 1993. Generalised probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–60.

S. Buchholz, J. Veenstra, and W. Daelemans. 1999. Cascaded grammatical relation assignment. In P. Fung and J. Zhou, editors, *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC)*, pages 239–246.

S. Buchholz. 2002. *Memory-Based Grammatical Relation Finding.* Ph.D. thesis, University of Tilburg.

J. Carroll, E. Briscoe, and A. Sanfilippo. 1998. Parser evaluation: A survey and a new proposal. In *Proceedings of the International Conference on Language Resources and Evaluation*, pages 447–454.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL-2000*, pages 132–139.

S. Clark and J. Hockenmaier. 2002. Evaluating a wide-coverage CCG parser. In *Proceedings of the LREC 2002 Beyond Parseval Workshop.*

M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the ACL (jointly with the 8th Conference of the EACL)*, pages 16–23.

W. Daelemans, J. Zavrel, P. Berck, and S. Gillis. 1996. MBT: A memory-based part of speech tagger generator. In E. Ejerhed and I. Dagan, editors, *Proceedings of the 4th ACL/SIGDAT Workshop on Very Large Corpora*, pages 14–27.

W. Daelemans. 1996. Abstraction considered harmful: Lazy learning of language processing. In H. J. van den Herik and A. Weijiters, editors, *Proceedings of the Sixth Beligian-Dutch Conference on Machine Learning*, pages 3–12.

D. Elworthy. 1994. Does Baum-Welch re-estimation help taggers? In *Proceedings of the 4th Conference on Applied NLP*, pages 53–58.

S. Lappin and H. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.

G. Leech. 1992. 100 million words of English: the British National Corpus. *Language Research*, 28(1):1–13.

D. Lin. 1995. Dependency-based parser evaluation: a study with a software manual corpus. In R. Sutcliffe, H-D. Koch, and A. McEllingott, editors, *Industrial Parsing of Software Manuals*, pages 13–24.

M. Marcus, R. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn TreeBank. *Computational Linguisitcs*, 19(2):313–330.

J. Preiss and E. Briscoe. 2003. Shallow or full parsing for anaphora resolution? An experiment with the Lappin and Leass algorithm. In *Proceedings of the Workshop on Anaphora Resolution.*

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142.

G. Sampson. 1995. *English for the computer.* Oxford University Press.

J. Veenstra and A. van den Bosch. 2000. Single-classifier memory-based phrase chunking. In *Proceedings of the Fourth Conference on Computational Natural Language Learning (CoNLL) and the Second Learning Language in Logic Workshop (LLL)*, pages 157–159.