Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations

Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, Andy Way
National Centre for Language Technology and School of Computing,
Dublin City University, Dublin, Ireland
{acahill,mburke,rodonovan,josef,away}@computing.dcu.ie

Abstract

This paper shows how finite approximations of long distance dependency (LDD) resolution can be obtained automatically for wide-coverage, robust, probabilistic Lexical-Functional Grammar (LFG) resources acquired from treebanks. We extract LFG subcategorisation frames and paths linking LDD reentrancies from f-structures generated automatically for the Penn-II treebank trees and use them in an LDD resolution algorithm to parse new text. Unlike (Collins, 1999; Johnson, 2002), in our approach resolution of LDDs is done at f-structure (attribute-value structure representations of basic predicate-argument or dependency structure) without empty productions, traces and coindexation in CFG parse trees. Currently our best automatically induced grammars achieve 80.97% f-score for f-structures parsing section 23 of the WSJ part of the Penn-II treebank and evaluating against the DCU 105[1] and 80.24% against the PARC 700 Dependency Bank (King et al., 2003), performing at the same or a slightly better level than state-of-the-art hand-crafted grammars (Kaplan et al., 2004).

1 Introduction

The determination of syntactic structure is an important step in natural language processing as syntactic structure strongly determines semantic interpretation in the form of predicate-argument structure, dependency relations or logical form. For a substantial number of linguistic phenomena such as topicalisation, wh-movement in relative clauses and interrogative sentences, however, there is an important difference between the location of the (surface) realisation of linguistic material and the location where this material should be interpreted semantically. Resolution of such long-distance dependencies (LDDs) is therefore crucial in the determination of accurate predicate-argument structure, deep dependency relations and the construction of proper meaning representations such as logical forms (Johnson, 2002).

Modern unification/constraint-based grammars such as LFG or HPSG capture deep linguistic information including LDDs, predicate-argument structure, or logical form. Manually scaling rich unification grammars to naturally occurring free text, however, is extremely time-consuming, expensive and requires considerable linguistic and computational expertise. Few hand-crafted, deep unification grammars have in fact achieved the coverage and robustness required to parse a corpus of say the size and complexity of the Penn treebank: (Riezler et al., 2002) show how a deep, carefully hand-crafted LFG is successfully scaled to parse the Penn-II treebank (Marcus et al., 1994) with discriminative (log-linear) parameter estimation techniques.

The last 20 years have seen continuously increasing efforts in the construction of parse-annotated corpora. Substantial treebanks[2] are now available for many languages (including English, Japanese, Chinese, German, French, Czech, Turkish), others are currently under construction (Arabic, Bulgarian) or near completion (Spanish, Catalan). Treebanks have been enormously influential in the development of robust, state-of-the-art parsing technology: grammars (or grammatical information) automatically extracted from treebank resources provide the backbone of many state-of-the-art probabilistic parsing approaches (Charniak, 1996; Collins, 1999; Charniak, 1999; Hockenmaier, 2003; Klein and Manning, 2003). Such approaches are attractive as they achieve robustness, coverage and performance while incurring very low grammar development cost. However, with few notable exceptions (e.g. Collins' Model 3, (Johnson, 2002), (Hockenmaier, 2003)), treebank-based probabilistic parsers return fairly simple "surfacey" CFG trees, without deep syntactic or semantic information. The grammars used by such systems are sometimes re-

[1]Manually constructed f-structures for 105 randomly selected trees from Section 23 of the WSJ section of the Penn-II Treebank

[2]Or dependency banks.

ferred to as “half” (or “shallow”) grammars (Johnson, 2002), i.e. they do not resolve LDDs but interpret linguistic material purely locally where it occurs in the tree.

Recently (Cahill et al., 2002) showed how wide-coverage, probabilistic unification grammar resources can be acquired automatically from f-structure-annotated treebanks. Many second generation treebanks provide a certain amount of deep syntactic or dependency information (e.g. in the form of Penn-II functional tags and traces) supporting the computation of representations of deep linguistic information. Exploiting this information (Cahill et al., 2002) implement an automatic LFG f-structure annotation algorithm that associates nodes in treebank trees with f-structure annotations in the form of attribute-value structure equations representing abstract predicate-argument structure/dependency relations. From the f-structure annotated treebank they automatically extract wide-coverage, robust, PCFG-based LFG approximations that parse new text into trees and f-structure representations.

The LFG approximations of (Cahill et al., 2002), however, are only “half” grammars, i.e. like most of their probabilistic CFG cousins (Charniak, 1996; Johnson, 1999; Klein and Manning, 2003) they do not resolve LDDs but interpret linguistic material purely locally where it occurs in the tree.

In this paper we show how finite approximations of long distance dependency resolution can be obtained automatically for wide-coverage, robust, probabilistic LFG resources automatically acquired from treebanks. We extract LFG subcategorisation frames and paths linking LDD reentrancies from f-structures generated automatically for the Penn-II treebank trees and use them in an LDD resolution algorithm to parse new text. Unlike (Collins, 1999; Johnson, 2002), in our approach LDDs are resolved on the level of f-structure representation, rather than in terms of empty productions and co-indexation on parse trees. Currently we achieve f-structure/dependency f-scores of 80.24 and 80.97 for parsing section 23 of the WSJ part of the Penn-II treebank, evaluating against the PARC 700 and DCU 105 respectively.

The paper is structured as follows: we give a brief introduction to LFG. We outline the automatic f-structure annotation algorithm, PCFG-based LFG grammar approximations and parsing architectures of (Cahill et al., 2002). We present our subcategorisation frame extraction and introduce the treebank-based acquisition of finite approximations of LFG functional uncertainty equations in terms of LDD paths. We present the f-structure LDD resolution algorithm, provide results and extensive evaluation. We compare our method with previous work. Finally, we conclude.

2 Lexical Functional Grammar (LFG)

Lexical-Functional Grammar (Kaplan and Bresnan, 1982; Dalrymple, 2001) minimally involves two levels of syntactic representation:[3] c-structure and f-structure. C(onstituent)-structure represents the grouping of words and phrases into larger constituents and is realised in terms of a CF-PSG grammar. F(unctional)-structure represents abstract syntactic functions such as SUBJ(ect), OBJ(ect), OBL(ique), closed and open clausal COMP/XCOMP(lement), ADJ(unct), APP(osition) etc. and is implemented in terms of recursive feature structures (attribute-value matrices). C-structure captures surface grammatical configurations, f-structure encodes abstract syntactic information approximating to predicate-argument/dependency structure or simple logical form (van Genabith and Crouch, 1996). C- and f-structures are related in terms of functional annotations (constraints, attribute-value equations) on c-structure rules (cf. Figure 1).

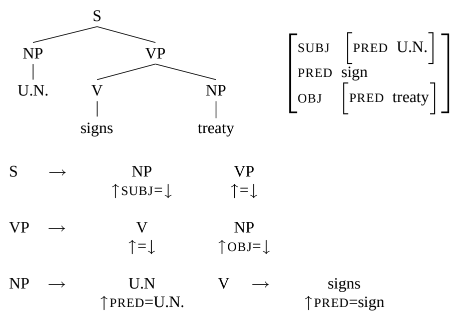

Figure 1: Simple LFG C- and F-Structure

Uparrows point to the f-structure associated with the mother node, downarrows to that of the local node. The equations are collected with arrows instantiated to unique tree node identifiers, and a constraint solver generates an f-structure.

3 Automatic F-Structure Annotation

The Penn-II treebank employs CFG trees with additional “functional” node annotations (such as -LOC, -TMP, -SBJ, -LGS, . . .) as well as traces and coindexation (to indicate LDDs) as basic data structures. The f-structure annotation algorithm of (Cahill et

[3]LFGs may also involve morphological and semantic levels of representation.

al., 2002) exploits configurational, categorial, Penn-II "functional", local head and trace information to annotate nodes with LFG feature-structure equations. A slightly adapted version of (Magerman, 1994)'s scheme automatically head-lexicalises the Penn-II trees. This partitions local subtrees of depth one (corresponding to CFG rules) into left and right contexts (relative to head). The annotation algorithm is modular with four components (Figure 2): left-right (L-R) annotation principles (e.g. leftmost NP to right of V head of VP type rule is likely to be an object etc.); coordination annotation principles (separating these out simplifies other components of the algorithm); traces (translates traces and coindexation in trees into corresponding reentrancies in f-structure ($\boxed{1}$ in Figure 3)); catch all and clean-up. Lexical information is provided via macros for POS tag classes.

$$\begin{tikzcd}\boxed{\text{L/R Context}} \arrow[r, Rightarrow] & \boxed{\text{Coordination}} \arrow[r, Rightarrow] & \boxed{\text{Traces}} \arrow[r, Rightarrow] & \boxed{\text{Catch-All}}\end{tikzcd}$$

Figure 2: Annotation Algorithm

The f-structure annotations are passed to a constraint solver to produce f-structures. Annotation is evaluated in terms of coverage and quality, summarised in Table 1. Coverage is near complete with 99.82% of the 48K Penn-II sentences receiving a single, connected f-structure. Annotation quality is measured in terms of precision and recall (P&R) against the DCU 105. The algorithm achieves an F-score of 96.57% for full f-structures and 94.3% for preds-only f-structures.[4]

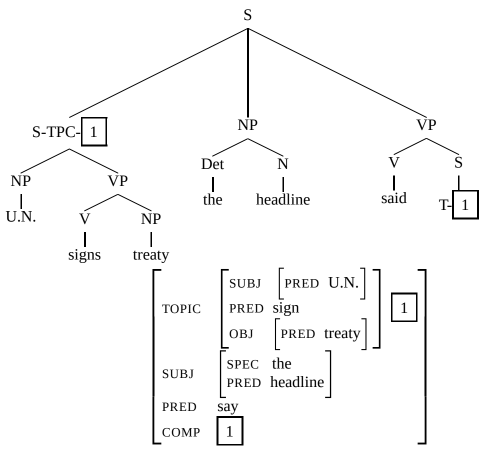

Figure 3: Penn-II style tree with LDD trace and corresponding reentrancy in f-structure

[4]Full f-structures measure all attribute-value pairs including"minor" features such as person, number etc. The stricter preds-only captures only paths ending in PRED:VALUE.

# frags	# sent	percent
0	85	0.176
1	48337	99.820
2	2	0.004

	all	preds
P	96.52	94.45
R	96.63	94.16

Table 1: F-structure annotation results for DCU 105

4 PCFG-Based LFG Approximations

Based on these resources (Cahill et al., 2002) developed two parsing architectures. Both generate PCFG-based approximations of LFG grammars.

In the **pipeline** architecture a standard PCFG is extracted from the "raw" treebank to parse unseen text. The resulting parse-trees are then annotated by the automatic f-structure annotation algorithm and resolved into f-structures.

In the **integrated** architecture the treebank is first annotated with f-structure equations. An annotated PCFG is then extracted where each non-terminal symbol in the grammar has been augmented with LFG f-equations: $\mathrm{NP}[\uparrow\textsc{obj}{=}\downarrow] \rightarrow \mathrm{DT}[\uparrow\textsc{spec}{=}\downarrow]\ \mathrm{NN}[\uparrow{=}\downarrow]$. Nodes followed by annotations are treated as a monadic category for grammar extraction and parsing. Post-parsing, equations are collected from parse trees and resolved into f-structures.

Both architectures parse raw text into "proto" f-structures with LDDs unresolved resulting in incomplete argument structures as in Figure 4.

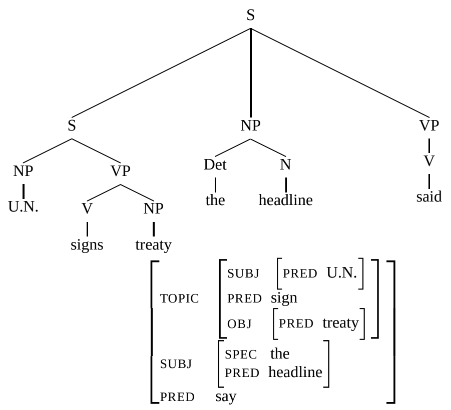

Figure 4: Shallow-Parser Output with Unresolved LDD and Incomplete Argument Structure (cf. Figure 3)

5 LDDs and LFG FU-Equations

Theoretically, LDDs can span unbounded amounts of intervening linguistic material as in

[U.N. signs treaty]$_1$ the paper claimed ... a source said []$_1$.

In LFG, LDDs are resolved at the f-structure level, obviating the need for empty productions and traces in trees (Dalrymple, 2001), using functional uncertainty (FU) equations. FUs are regular expressions specifying paths in f-structure between a source (where linguistic material is encountered) and a target (where linguistic material is interpreted semantically). To account for the fronted sentential constituents in Figures 3 and 4, an FU equation of the form $\uparrow$ TOPIC = $\uparrow$ COMP* COMP would be required. The equation states that the value of the TOPIC attribute is token identical with the value of the final COMP argument along a path through the immediately enclosing f-structure along zero or more COMP attributes. This FU equation is annotated to the topicalised sentential constituent in the relevant CFG rules as follows

$$\begin{array}{llll} \text{S} \rightarrow & \text{S} & \text{NP} & \text{VP} \\ & \uparrow\text{TOPIC}=\downarrow & \uparrow\text{SUBJ}=\downarrow & \uparrow=\downarrow \\ & \uparrow\text{TOPIC}=\uparrow\text{COMP*COMP} & & \end{array}$$

and generates the LDD-resolved proper f-structure in Figure 3 for the traceless tree in Figure 4, as required.

In addition to FU equations, subcategorisation information is a crucial ingredient in LFG's account of LDDs. As an example, for a topicalised constituent to be resolved as the argument of a local predicate as specified by the FU equation, the local predicate must (i) subcategorise for the argument in question and (ii) the argument in question must not be already filled. Subcategorisation requirements are provided lexically in terms of semantic forms (subcat lists) and coherence and completeness conditions (all GFs specified must be present, and no others may be present) on f-structure representations. Semantic forms specify which grammatical functions (GFs) a predicate requires locally. For our example in Figures 3 and 4, the relevant lexical entries are:

$$\begin{array}{llll} \text{V} & \rightarrow & \text{said} & \uparrow\text{PRED=say}\langle\uparrow \text{SUBJ}, \uparrow \text{COMP}\rangle \\ \text{V} & \rightarrow & \text{signs} & \uparrow\text{PRED=sign}\langle\uparrow \text{SUBJ}, \uparrow \text{OBJ}\rangle \end{array}$$

FU equations and subcategorisation requirements together ensure that LDDs can only be resolved at suitable f-structure locations.

6 Acquiring Lexical and LDD Resources

In order to model the LFG account of LDD resolution we require subcat frames (i.e. semantic forms) and LDD resolution paths through f-structure. Traditionally, such resources were handcoded. Here we show how they can be acquired from f-structure annotated treebank resources.

LFG distinguishes between governable (arguments) and nongovernable (adjuncts) grammatical functions (GFs). If the automatic f-structure annotation algorithm outlined in Section 3 generates high quality f-structures, reliable semantic forms can be extracted (reverse-engineered): for each f-structure generated, for each level of embedding we determine the local PRED value and collect the governable, i.e. subcategorisable grammatical functions present at that level of embedding. For the proper f-structure in Figure 3 we obtain `sign([subj,obj])` and `say([subj,comp])`. We extract frames from the full WSJ section of the Penn-II Treebank with 48K trees. Unlike many other approaches, our extraction process does not predefine frames, fully reflects LDDs in the source data-structures (cf. Figure 3), discriminates between active and passive frames, computes GF, GF:CFG category pair- as well as CFG category-based subcategorisation frames and associates conditional probabilities with frames. Given a lemma l and an argument list s, the probability of s given l is estimated as:

$$\mathcal{P}(s|l) := \frac{count(l,s)}{\sum_{i=1}^{n} count(l,s_i)}$$

Table 2 summarises the results. We extract 3586 verb lemmas and 10969 unique verbal semantic form types (lemma followed by non-empty argument list). Including prepositions associated with the subcategorised OBLs and particles, this number goes up to 14348. The number of unique frame types (without lemma) is 38 without specific prepositions and particles, 577 with. F-structure annotations allow us to distinguish passive and active frames. Table 3 shows the most frequent semantic forms for `accept`. Passive frames are marked **p**. We carried out a comprehensive evaluation of the automatically acquired verbal semantic forms against the COMLEX Resource (Macleod et al., 1994) for the 2992 active verb lemmas that both resources have in common. We report on the evaluation of GF-based frames for the full frames with complete prepositional and particle infomation. We use relative conditional probability thresholds (1% and 5%) to filter the selection of semantic forms (Table 4). (O'Donovan et al., 2004) provide a more detailed description of the extraction and evaluation of semantic forms.

	Without Prep/Part	**With Prep/Part**
Lemmas	3586	3586
Sem. Forms	10969	14348
Frame Types	38	577
Active Frame Types	38	548
Passive Frame Types	21	177

Table 2: Verb Results

Semantic Form	Occurrences	Prob.
`accept([obj,subj])`	122	0.813
`accept([subj],`**`p`**`)`	9	0.060
`accept([comp,subj])`	5	0.033
`accept([subj,obl:as],`**`p`**`)`	3	0.020
`accept([obj,subj,obl:as])`	3	0.020
`accept([obj,subj,obl:from])`	3	0.020
`accept([subj])`	2	0.013
`accept([obj,subj,obl:at])`	1	0.007
`accept([obj,subj,obl:for])`	1	0.007
`accept([obj,subj,xcomp])`	1	0.007

Table 3: Semantic forms for the verb `accept`.

	Threshold 1%			Threshold 5%		
	P	**R**	**F-Score**	**P**	**R**	**F-Score**
Exp.	73.7%	22.1%	34.0%	78.0%	18.3%	29.6%

Table 4: COMLEX Comparison

We further acquire *finite approximations* of FU-equations. by extracting paths between co-indexed material occurring in the automatically generated f-structures from sections 02-21 of the Penn-II treebank. We extract 26 unique TOPIC, 60 TOPIC-REL and 13 FOCUS path types (with a total of 14,911 token occurrences), each with an associated probability. We distinguish between two types of TOPIC-REL paths, those that occur in wh-less constructions, and all other types (c.f Table 5). Given a path p and an LDD type t (either TOPIC, TOPIC-REL or FOCUS), the probability of p given t is estimated as:

$$\mathcal{P}(p|t) := \frac{count(t,p)}{\sum_{i=1}^{n} count(t,p_i)}$$

In order to get a first measure of how well the approximation models the data, we compute the path types in section 23 not covered by those extracted from 02-21: 23/(02-21). There are 3 such path types (Table 6), each occuring exactly once. Given that the total number of path tokens in section 23 is 949, the finite approximation extracted from 02-23 covers 99.69% of all LDD paths in section 23.

7 Resolving LDDs in F-Structure

Given a set of semantic forms s with probabilities $\mathcal{P}(s|l)$ (where l is a lemma), a set of paths p with $\mathcal{P}(p|t)$ (where t is either TOPIC, TOPIC-REL or FOCUS) and an f-structure f, the core of the algorithm to resolve LDDs recursively traverses f to:

```
find TOPIC|TOPIC-REL|FOCUS:$g$ pair; retrieve TOPIC|TOPIC-REL|FOCUS paths; for each path $p$ with GF$_1$ : ... : GF$_n$ : GF, traverse $f$ along GF$_1$ : ...: GF$_n$ to sub-f-structure $h$; retrieve local PRED:$l$;
    add GF:$g$ to $h$ iff
        ∗ GF is not present at $h$
        ∗ $h$ together with GF is locally complete and coherent with respect to a semantic form $s$ for $l$
    rank resolution by $\mathcal{P}(s|l) \times \mathcal{P}(p|t)$
```

wh-less TOPIC-REL	#	wh-less TOPIC-REL	#
subj	5692	adjunct	1314
xcomp:adjunct	610	obj	364
xcomp:obj	291	xcomp:xcomp:adjunct	96
comp:subj	76	xcomp:subj	67

Table 5: Most frequent wh-less TOPIC-REL paths

	02–21	23	23 /(02–21)
TOPIC	26	7	2
FOCUS	13	4	0
TOPIC-REL	60	22	1

Table 6: Number of path types extracted

The algorithm supports multiple, interacting TOPIC, TOPIC-REL and FOCUS LDDs. We use $\mathcal{P}(s|l) \times \mathcal{P}(p|t)$ to rank a solution, depending on how likely the PRED takes semantic frame s, and how likely the TOPIC, FOCUS or TOPIC-REL is resolved using path p. The algorithm also supports resolution of LDDs where no overt linguistic material introduces a source TOPIC-REL function (e.g. in reduced relative clause constructions). We distinguish between passive and active constructions, using the relevant semantic frame type when resolving LDDs.

8 Experiments and Evaluation

We ran experiments with grammars in both the pipeline and the integrated parsing architectures. The first grammar is a basic PCFG, while A-PCFG includes the f-structure annotations. We apply a parent transformation to each grammar (Johnson, 1999) to give P-PCFG and PA-PCFG. We train on sections 02-21 (grammar, lexical extraction and LDD paths) of the Penn-II Treebank and test on section 23. The only pre-processing of the trees that we do is to remove empty nodes, and remove all Penn-II functional tags in the integrated model. We evaluate the parse trees using evalb. Following (Riezler et al., 2002), we convert f-structures into dependency triple format. Using their software we evaluate the f-structure parser output against:

1. The DCU 105 (Cahill et al., 2002)

2. The full 2,416 f-structures *automatically* generated by the f-structure annotation algorithm for the *original* Penn-II trees, in a CCG-style (Hockenmaier, 2003) evaluation experiment

	Pipeline		Integrated	
	PCFG	P-PCFG	A-PCFG	PA-PCFG
	2416 Section 23 trees			
# Parses	2416	2416	2416	2414
Lab. F-Score	75.83	80.80	79.17	81.32
Unlab. F-Score	78.28	82.70	81.49	83.28
	DCU 105 F-Strs			
All GFs F-Score (before LDD resolution)	79.82	79.24	81.12	81.20
All GFs F-Score (after LDD resolution)	83.79	84.59	86.30	87.04
Preds only F-Score (before LDD resolution)	70.00	71.57	73.45	74.61
Preds only F-Score (after LDD resolution)	73.78	77.43	78.76	**80.97**
	2416 F-Strs			
All GFs F-Score (before LDD resolution)	81.98	81.49	83.32	82.78
All GFs F-Score (after LDD resolution)	84.16	84.37	86.45	86.00
Preds only F-Score (before LDD resolution)	72.00	73.23	75.22	75.10
Preds only F-Score (after LDD resolution)	74.07	76.12	78.36	**78.40**
	PARC 700 Dependency Bank			
Subset of GFs following (Kaplan et al., 2004)	77.86	**80.24**	77.68	78.60

Table 7: Parser Evaluation

3. A subset of 560 dependency structures of the PARC 700 Dependency Bank following (Kaplan et al., 2004)

The results are given in Table 7. The parent-transformed grammars perform best in both architectures. In all cases, there is a marked improvement (2.07-6.36%) in the f-structures after LDD resolution. We achieve between 73.78% and 80.97% preds-only and 83.79% to 87.04% all GFs f-score, depending on gold-standard. We achieve between 77.68% and 80.24% against the PARC 700 following the experiments in (Kaplan et al., 2004). For details on how we map the f-structures produced by our parsers to a format similar to that of the PARC 700 Dependency Bank, see (Burke et al., 2004). Table 8 shows the evaluation result broken down by individual GF (preds-only) for the integrated model PA-PCFG against the DCU 105. In order to measure how many of the LDD reentrancies in the gold-standard f-structures are captured correctly by our parsers, we developed evaluation software for f-structure LDD reentrancies (similar to Johnson's (2002) evaluation to capture traces and their antecedents in trees). Table 9 shows the results with the integrated model achieving more than 76% correct LDD reentrancies.

DEP.	PRECISION	RECALL	F-SCORE
adjunct	717/903 = 79	717/947 = 76	78
app	14/15 = 93	14/19 = 74	82
comp	35/43 = 81	35/65 = 54	65
coord	109/143 = 76	109/161 = 68	72
det	253/264 = 96	253/269 = 94	95
focus	1/1 = 100	1/1 = 100	100
obj	387/445 = 87	387/461 = 84	85
obj2	0/1 = 0	0/2 = 0	0
obl	27/56 = 48	27/61 = 44	46
obl2	1/3 = 33	1/2 = 50	40
obl_ag	5/11 = 45	5/12 = 42	43
poss	69/73 = 95	69/81 = 85	90
quant	40/55 = 73	40/52 = 77	75
relmod	26/38 = 68	26/50 = 52	59
subj	330/361 = 91	330/414 = 80	85
topic	12/12 = 100	12/13 = 92	96
topicrel	35/42 = 83	35/52 = 67	74
xcomp	139/160 = 87	139/146 = 95	91
OVERALL	83.78	78.35	80.97

Table 8: Preds-only results of PA-PCFG against the DCU 105

9 Related Work

(Collins, 1999)'s Model 3 is limited to wh-traces in relative clauses (it doesn't treat topicalisation, focus etc.). Johnson's (2002) work is closest to ours in spirit. Like our approach he provides a finite approximation of LDDs. Unlike our approach, however, he works with tree fragments in a post-processing approach to add empty nodes and their antecedents to parse trees, while we present an approach to LDD resolution on the level of f-structure. It seems that the f-structure-based approach is more abstract (99 LDD path types against approximately 9,000 tree-fragment types in (Johnson, 2002)) and fine-grained in its use of lexical information (subcat frames). In contrast to Johnson's approach, our LDD resolution algorithm is not biased. It computes all possible complete resolutions and order-ranks them using LDD path and subcat frame probabilities. It is difficult to provide a satisfactory comparison between the two methods, but we have carried out an experiment that compares them at the f-structure level. We take the output of Charniak's

	Pipeline		Integrated	
	PCFG	P-PCFG	A-PCFG	PA-PCFG
	TOPIC			
Precision	(11/14)	(12/13)	(12/13)	(12/12)
Recall	(11/13)	(12/13)	(12/13)	(12/13)
F-Score	0.81	0.92	0.92	0.96
	FOCUS			
Precision	(0/1)	(0/1)	(0/1)	(0/1)
Recall	(0/1)	(0/1)	(0/1)	(0/1)
F-Score	0	0	0	0
	TOPIC-REL			
Precision	(20/34)	(27/36)	(34/42)	(34/42)
Recall	(20/52)	(27/52)	(34/52)	(34/52)
F-Score	0.47	0.613	0.72	0.72
OVERALL	0.54	0.67	0.75	0.76

Table 9: LDD Evaluation on the DCU 105

Charniak	-LDD res.	+LDD res.	(Johnson, 2002)
All GFs	80.86	86.65	85.16
Preds Only	74.63	80.97	79.75

Table 10: Comparison at f-structure level of LDD resolution to (Johnson, 2002) on the DCU 105

parser (Charniak, 1999) and, using the pipeline f-structure annotation model, evaluate against the DCU 105, both before and after LDD resolution. Using the software described in (Johnson, 2002) we add empty nodes to the output of Charniak's parser, pass these trees to our automatic annotation algorithm and evaluate against the DCU 105. The results are given in Table 10. Our method of resolving LDDs at f-structure level results in a preds-only f-score of 80.97%. Using (Johnson, 2002)'s method of adding empty nodes to the parse-trees results in an f-score of 79.75%.

(Hockenmaier, 2003) provides CCG-based models of LDDs. Some of these involve extensive clean-up of the underlying Penn-II treebank resource prior to grammar extraction. In contrast, in our approach we leave the treebank as is and only add (but never correct) annotations. Earlier HPSG work (Tateisi et al., 1998) is based on independently constructed *hand-crafted* XTAG resources. In contrast, we acquire our resources from treebanks and achieve substantially wider coverage.

Our approach provides wide-coverage, robust, and – with the addition of LDD resolution – "deep" or "full", PCFG-based LFG *approximations*. Crucially, we do not claim to provide fully adequate statistical models. It is well known (Abney, 1997) that PCFG-type approximations to unification grammars can yield inconsistent probability models due to loss of probability mass: the parser successfully returns the highest ranked parse tree but the constraint solver cannot resolve the f-equations (generated in the pipeline or "hidden" in the integrated model) and the probability mass associated with that tree is lost. This case, however, is surprisingly rare for our grammars: only 0.0018% (85 out of 48424) of the original Penn-II trees (without FRAGs) fail to produce an f-structure due to inconsistent annotations (Table 1), and for parsing section 23 with the integrated model (A-PCFG), only 9 sentences do not receive a parse because no f-structure can be generated for the highest ranked tree (0.4%). Parsing with the pipeline model, all sentences receive one complete f-structure. Research on adequate probability models for unification grammars is important. (Miyao et al., 2003) present a Penn-II treebank based HPSG with log-linear probability models. They achieve coverage of 50.2% on section 23, as against 99% in our approach. (Riezler et al., 2002; Kaplan et al., 2004) describe how a carefully hand-crafted LFG is scaled to the full Penn-II treebank with log-linear based probability models. They achieve 79% coverage (full parse) and 21% fragement/skimmed parses. By the same measure, full parse coverage is around 99% for our automatically acquired PCFG-based LFG approximations. Against the PARC 700, the hand-crafted LFG grammar reported in (Kaplan et al., 2004) achieves an f-score of 79.6%. For the same experiment, our best automatically-induced grammar achieves an f-score of 80.24%.

10 Conclusions

We presented and extensively evaluated a finite approximation of LDD resolution in automatically constructed, wide-coverage, robust, PCFG-based LFG approximations, effectively turning the "half"(or "shallow")-grammars presented in (Cahill et al., 2002) into "full" or "deep" grammars. In our approach, LDDs are resolved in f-structure, not trees. The method achieves a preds-only f-score of 80.97% for f-structures with the PA-PCFG in the integrated architecture against the DCU 105 and 78.4% against the 2,416 automatically generated f-structures for the original Penn-II treebank trees. Evaluating against the PARC 700 Dependency Bank, the P-PCFG achieves an f-score of 80.24%, an overall improvement of approximately 0.6% on the result reported for the best hand-crafted grammars in (Kaplan et al., 2004).

Acknowledgements

This research was funded by Enterprise Ireland Basic Research Grant SC/2001/186 and IRCSET.

References

S. Abney. 1997. Stochastic attribute-value grammars. *Computational Linguistics*, 23(4):597–618.

M. Burke, A. Cahill, R. O'Donovan, J. van Genabith, and A. Way 2004. The Evaluation of an Automatic Annotation Algorithm against the PARC 700 Dependency Bank. In *Proceedings of the Ninth International Conference on LFG*, Christchurch, New Zealand (to appear).

A. Cahill, M. McCarthy, J. van Genabith, and A. Way. 2002. Parsing with PCFGs and Automatic F-Structure Annotation. In Miriam Butt and Tracy Holloway King, editors, *Proceedings of the Seventh International Conference on LFG*, pages 76–95. CSLI Publications, Stanford, CA.

E. Charniak. 1996. Tree-Bank Grammars. In *AAAI/IAAI, Vol. 2*, pages 1031–1036.

E. Charniak. 1999. A Maximum-Entropy-Inspired Parser. Technical Report CS-99-12, Brown University, Providence, RI.

M. Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

M. Dalrymple. 2001. *Lexical-Functional Grammar*. San Diego, CA; London Academic Press.

J. Hockenmaier. 2003. Parsing with Generative models of Predicate-Argument Structure. In *Proceedings of the 41st Annual Conference of the Association for Computational Linguistics*, pages 359–366, Sapporo, Japan.

M. Johnson. 1999. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632.

M. Johnson. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 136–143, Philadelphia, PA.

R. Kaplan and J. Bresnan. 1982. Lexical Functional Grammar, a Formal System for Grammatical Representation. In *The Mental Representation of Grammatical Relations*, pages 173–281. MIT Press, Cambridge, MA.

R. Kaplan, S. Riezler, T. H. King, J. T. Maxwell, A. Vasserman, and R. Crouch. 2004. Speed and accuracy in shallow and deep stochastic parsing. In *Proceedings of the Human Language Technology Conference and the 4th Annual Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 97–104, Boston, MA.

T.H. King, R. Crouch, S. Riezler, M. Dalrymple, and R. Kaplan. 2003. The PARC700 dependency bank. In *Proceedings of the EACL03: 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*, pages 1–8, Budapest.

D. Klein and C. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 423–430, Sapporo, Japan.

C. Macleod, A. Meyers, and R. Grishman. 1994. The COMLEX Syntax Project: The First Year. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 669-703, Princeton, NJ.

D. Magerman. 1994. *Natural Language Parsing as Statistical Pattern Recognition*. PhD thesis, Stanford University, CA.

M. Marcus, G. Kim, M.A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 110–115, Princeton, NJ.

Y. Miyao, T. Ninomiya, and J. Tsujii. 2003. Probabilistic modeling of argument structures including non-local dependencies. In *Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 285–291, Borovets, Bulgaria.

R. O'Donovan, M. Burke, A. Cahill, J. van Genabith, and A. Way. 2004. Large-Scale Induction and Evaluation of Lexical Resources from the Penn-II Treebank. In *Proceedings of the 42nd Annual Conference of the Association for Computational Linguistics (ACL-04)*, Barcelona.

S. Riezler, T.H. King, R. Kaplan, R. Crouch, J. T. Maxwell III, and M. Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-02)*, pages 271–278, Philadelphia, PA.

Y. Tateisi, K. Torisawa, Y. Miyao, and J. Tsujii. 1998. Translating the XTAG English Grammar to HPSG. In *4th International Workshop on Tree Adjoining Grammars and Related Frameworks*, Philadelphia, PA, pages 172–175.

J. van Genabith and R. Crouch. 1996. Direct and Underspecified Interpretations of LFG f-Structures. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 262–267, Copenhagen.