Chapter 25

Treebank-driven Parsing, Translation and Grammar Induction using LFG

Aoife Cahill
Educational Testing Service

Andy Way
ADAPT Centre, School of Computing, Dublin City University

This chapter provides a summary of a range of work on probabilistic models of Lexical Functional Grammar (LFG). LFG grammars as originally conceived in Kaplan & Bresnan (1982) were defined by grammatical rules and constraints, so could not describe ill-formed strings, and they failed if confronted with well-formed strings outside their coverage. In contrast, the hybrid LFG-DOP model of Bod & Kaplan 1998 and Bod & Kaplan 2003 could generalize well-formed analyses via the *Discard* operation to allow ill-formed and previously uncovered well-formed strings to be handled.

Way (1999) and Way (2001) extended LFG-DOP to handle translation, and demonstrated two advantages of his LFG-DOT models: (i) being probabilistic, LFG-DOT was able to handle a range of translation phenomena that were problematic for the description of LFG-MT (Kaplan et al. 1989); and (ii) having f-structure constraints enabled LFG-DOT to overcome problems for DOT (Poutsma 2000), a model of translation based on DOP (Bod 1992; Sima'an 1997; Bod 1998).

Like most probabilistic models, LFG-DOP (and LFG-DOT) require large amounts of annotated data. In a range of seminal work on grammar induction – now a research field in its own right, but at the time quite a novelty – it was demonstrated how strings could be automatically annotated with both LFG c- and f-structure information (Sadler et al. 2000; Cahill et al. 2002a). These were then used for multilingual probabilistic parsing (Cahill et al. 2005; Cahill, Burke, O'Donovan, Riezler, et al. 2008) and lexicon induction experiments (O'Donovan 2006), which we describe here.

Aoife Cahill and Andy Way. Treebank-driven Parsing, Translation and Grammar Induction using LFG. To appear in: Dalrymple, Mary (ed.). Handbook of Lexical Functional Grammar. Berlin: Language Science Press. [preliminary page numbering]

1 Introduction

In this chapter we summarize work on extensions to the core LFG formalism that facilitate large-scale probabilistic LFG parsing and translation models. Traditional LFG grammars (Kaplan & Bresnan 1982) are defined in terms of well-formed grammatical rules and constraints. This has two main limitations: (i) ill-formed input cannot be handled easily;[1] and (ii) when a grammar produces multiple analyses for an input, there is no inherent way of ranking the competing solutions.

We describe LFG-DOP (Bod & Kaplan 1998), a hybrid model of Data-Oriented Parsing (DOP: Bod 1992; Sima'an 1997; Bod 1998) and LFG that allows for probabilistic tree parsing, and which is beyond context-free in its generative power. We describe how this work led to the LFG-DOT framework (Way 1999, 2001) for machine translation (MT) with LFG.

Large-scale probabilistic parsing typically requires substantial amounts of annotated training data. We describe techniques developed to automatically generate large-scale LFG-annotated treebanks that provide the training data needed for probabilistic LFG parsing. We describe how this work was not only applied to English, but also several other languages including German (Cahill et al. 2005; Rehbein & van Genabith 2009), French (Schluter 2011), Spanish (O'Donovan et al. 2005; Chrupała & van Genabith 2006), Chinese (Burke, Cahill, et al. 2004; Guo 2009), Japanese (Oya & van Genabith 2007) and Arabic (Tounsi et al. 2009a). A related field of work was the automated extraction of large-scale lexical resources from these LFG-annotated treebanks (O'Donovan 2006). Although large-scale LFG-DOT experimentation has not been conducted to date,[2] these grammars and semantic forms (i.e. subcategorisation frames) are exactly what LFG-DOT requires to build its models. Accordingly, we sketch what would need to be done to conduct such experiments.

Finally, we compare this semi-automatic approach to lexicon and grammar induction to that based on the hand-crafted XLE grammars.

2 LFG-DOP

This section describes how LFG was combined with Data-Oriented Parsing (DOP) models to create a more robust, probabilistic model of language processing, LFG-

[1] This applies equally to legitimate strings which are not covered by the grammar.

[2] Bod (2000) acknowledges that Cormons (1999) "accomplished [the] first simple experiment with LFG-DOP". Bod & Kaplan (2003) includes a large-scale evaluation of LFG-DOP against a DOP baseline. Hearne (2005) extends these experiments for DOP, demonstrating higher accuracy for the exact match metric using improved sampling techniques.

DOP. In later sections, we will show how both DOP and LFG-DOP were used to build powerful, robust models of MT.

2.1 Data-Oriented Parsing

DOP models (Bod 1992; Sima'an 1997; Bod 1998) assume that past experiences of language are significant in both perception and production. DOP prefers performance models over competence grammars, in that abstract grammar rules are eschewed in favour of models based on large collections of previously occurring fragments of language. Previously uncovered sentences are processed with reference to existing fragments from the treebank, which are combined using probabilistic techniques to determine the most likely analysis for the new fragment.

The general DOP architecture stipulates four parameters on which particular models are instantiated:

1. A formal definition of *well-formed representations* for sentence analyses;
2. A set of *decomposition* operations for splitting sentence analyses into a set of fragments;
3. A set of *composition* operations for recombination of such fragments in order to derive analyses of new strings;
4. A definition of a *probability model* indicating the likelihood of a sentence analysis based on the probabilities of its constituent parts.

DOP models typically assign a surface phrase-structure (PS) tree to strings (hence 'Tree-DOP', or 'DOP1' in Bod (1992)). However, context-free models are insufficiently powerful to deal with all aspects of human language. LFG, on the other hand, is known to be beyond context-free, and can capture and provide representations of linguistic phenomena other than those occurring at surface structure.[3]

[3]Note that the question of what grammar type in the Chomsky Hierarchy (Chomsky 1956) was capable of processing human language was a significant one when LFG was first proposed, but appears to be less of a concern nowadays. This was relevant for Chomsky's claims of Universal Grammar (Chomsky 1981), of course, but different languages have been demonstrated to require different grammar types; for example, Dutch cross-serial dependencies can only be handled by a context-sensitive grammar, whereas English is arguably context-free. Note that Futrell et al. (2016) claim the Amazonian language Pirahã to be finite-state, so the Chomsky Hierarchy no longer seems to be particularly helpful as a characterisation of human languages in general. Nonetheless, the fact that LFG is beyond context-free would allow it to claim that it is a general enough model to cope with languages like Dutch. Note too that a grammar formalism should be sufficiently constrained to ensure that parsing can be done in polynomial time.

2.2 Combining DOP with LFG: LFG-DOP

Accordingly, Bod & Kaplan (1998) augmented DOP with the syntactic representations of LFG to create a new, more powerful hybrid model of language processing – LFG-DOP – which adds a level of robustness not available to models based solely on LFG.

LFG-DOP is defined using the same four parameters as in Tree-DOP. We describe each of these in the next sections.

2.2.1 Representations in LFG-DOP

The LFG-DOP **representations** are those traditionally used in LFG, where each string is annotated with a c-structure, an f-structure, and a mapping ϕ between them. Well-formedness conditions operate solely on f-structure, as usual.

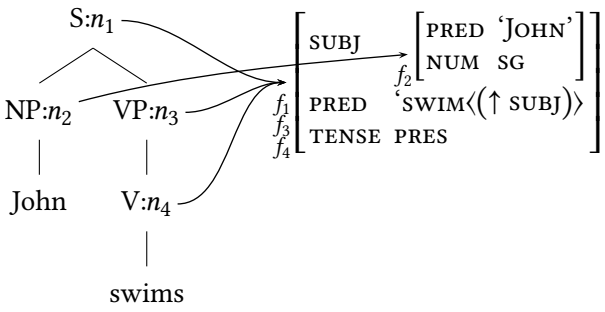
2.2.2 Decomposition in LFG-DOP

Since we are now dealing with $\langle c, f \rangle$ pairs of structure, the *Root* and *Frontier* **decomposition** operations of DOP need to be adapted to stipulate exactly which c-structure nodes are linked to which f-structure fragments, thereby maintaining the fundamentals of c- and f-structure correspondence. As LFG c-structures are little more than annotated PS trees, we can proceed very much on the same lines as in Tree-DOP. *Root* erases all nodes outside of the selected node, and in addition deletes all ϕ-links (informally, parts of the f-structure linked to a c-structure node) leaving the erased nodes, as well as all f-structure units that are not ϕ-accessible from the remaining nodes. Bod & Kaplan (1998) define ϕ-accessibility as follows:

> "An f-structure unit f is ϕ-accessible from a node n iff either n is ϕ-linked to f (that is, $f = \phi(n)$) or f is contained within $\phi(n)$ (that is, there is a chain of attributes that leads from $\phi(n)$ to f)." (Bod & Kaplan 1998: 146)

As an example, consider (1):

(1)



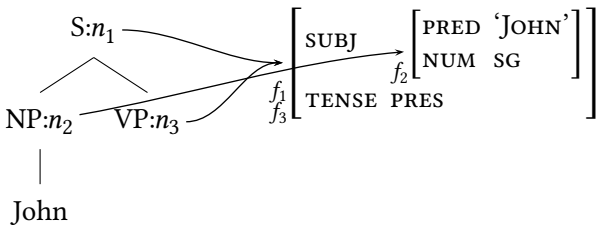
The ϕ-links are shown in (2):

(2) $\phi(n_1) = f_1, \phi(n_2) = f_2, \phi(n_3) = f_3, \phi(n_4) = f_4, \phi(n_1) = \phi(n_3) = \phi(n_4)$

ϕ-accessibility reflects the intuitive notion that nodes in a tree carry information only about the f-structure elements to which the root node of the tree permits access, as in (1). Note that all f-structure units are ϕ-accessible from the S, VP and V nodes, but TENSE and the top-level PRED (the main verb *swim*) cannot be accessed via ϕ from the subject NP node.

Frontier operates as in Tree-DOP, deleting all subtrees of the selected frontier nodes. It also deletes all ϕ-links of these deleted nodes together with any semantic form (e.g. in (1), 'SWIM⟨(↑ SUBJ)⟩') as is the case if the V:swims node is deleted in (3):

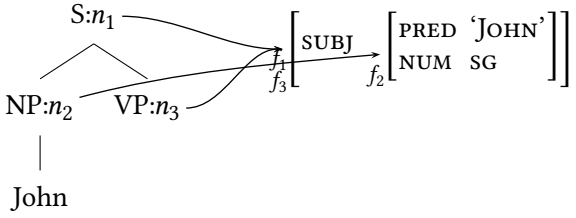
(3)



This illustrates the ability of *Root* nodes to access certain f-structure features even after subnodes have been deleted. Even though the V:swims node is deleted in the c-structure tree, only the semantic form 'swim⟨(↑ SUBJ)⟩' is deleted from the f-structure, and the TENSE feature remains.[4]

It is, however, possible to prune (3) still further, as (4) illustrates:

(4)



This is achieved by applying a third, and new operation, *Discard*, to the TENSE feature in (3).[5] The *Discard* operation adds considerably to LFG's robustness by

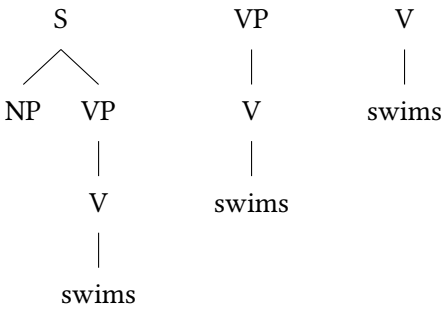
[4]Note that subject-tense agreement *is* seen in some languages e.g. Hindi. Accordingly, there is no universal principle which should rule out fragments such as (3).

[5]This function generates appropriate fragments for English which have no subject-tense dependency; accordingly, we would expect more fragments like (4) in English treebanks, but fewer such fragments for Hindi, say, given the point made in fn. 4.

providing generalized fragments from those derived via *Root* and *Frontier* by freely deleting any combination of attribute-value pairs from an f-structure except those that are ϕ-linked to some remaining c-structure node, or that are governed by the local predicate (i.e. required to be present). Its introduction also necessitates a new definition of the grammaticality of a sentence *with respect to a corpus*, namely any sentence having at least one derivation whose fragments are produced only by *Root* and *Frontier* and not by *Discard*. Way (1999) splits fragments into separate bags of *Discard* and non-*Discard* fragments in order "to facilitate the consideration of grammaticality." Bod (2000) demonstrates that this is helpful for LFG-DOP, too, on experiments with the Verbmobil and Homecentre corpora, which compare favourably with the original model of Bod & Kaplan (1998). In contrast, Hearne & Sima'an (2004) present an improved back-off estimation method where non-*Discard* fragments are naturally preferred.

We omit here the complete LFG-DOP treebank (ignoring the effects of the *Discard* operator) for the sentence *John swims*, but refer the interested reader to Figure 4.1 in Way (2001: 114). Nonetheless, as he does, we point out that each c-structure fragment in an LFG-DOP corpus is not necessarily linked to a unique f-structure fragment. From his Figure 4.1, consider the three fragments in (5):

(5)



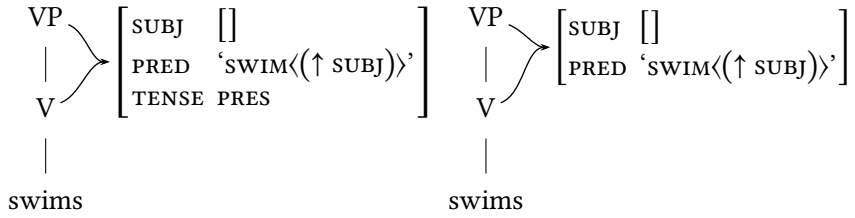
These three c-structure fragments all map to the same f-structure fragment in (6) because of equations such as $\phi(n_1) = \phi(n_3) = \phi(n_4)$ in (2):

(6)
$$\begin{bmatrix} \textsc{subj} & \left[\textsc{num}\ \textsc{sg}\right] \\ \textsc{pred} & \text{`}\textsc{swim}\langle(\uparrow \textsc{subj})\rangle\text{'} \\ \textsc{tense} & \textsc{pres} \end{bmatrix}$$

This f-structure shows that *swims* being singular requires a singular subject. Of course, to be completely accurate, we should add in a SUBJ:PERS:3 constraint too, to prevent strings such as *I swims* and *you swims* from being deemed grammatical.

We can illustrate the effect of *Discard* in relaxing the SUBJ:NUM:SG constraint with *swims* in (7):

(7)



Accordingly, if the ill-formed string *The men swims* were input, it could be processed by LFG-DOP because of generalised fragments like these, but would be ruled out as ungrammatical in LFG, given f-structures like (6). Note that *Discard* has been applied to the rightmost f-structure in (7).

2.2.3 Composition in LFG-DOP

Composition in LFG-DOP is also a two-step operation. C-structures are combined by leftmost substitution, as in Tree-DOP, subject to the matching of their nodes. F-structures corresponding to these nodes are then recursively unified, and the resulting f-structures are subjected to the grammaticality checks of LFG.

2.2.4 Probability models for LFG-DOP

$CP(f \mid CS)$ denotes the probability of choosing a fragment f from a competition set CS of competing fragments. In Tree-DOP, we wanted to select a tree t from a treebank, whereas in LFG-DOP we are interested in selecting a $\langle c, f \rangle$ pair from a corpus. The probability of an LFG-DOP derivation is the same as in Tree-DOP; it is just the derivation itself which changes. As in DOP, then, an LFG-DOP derivation $D = \langle f_1, f_2 ... f_n \rangle$ is produced by a stochastic branching process which at each stage in the process randomly samples from a competition set CS of competing samples, as in (25.8) (cf. example (10) in Bod & Kaplan 1998: 148):

$$P(\langle f_1, f_2 ... f_n \rangle) = \prod_{i=1}^{n} CP(f_i \mid CS_i) \tag{25.8}$$

This competition probability $CP(f \mid CS)$ is expressed in terms of fragment probabilities $P(f)$ in (25.9) (cf. example (11) in Bod & Kaplan 1998: 148):

$$CP(f \mid CS) = \frac{P(f)}{\sum_{f' \in CS} P(f')} \tag{25.9}$$

Taking (25.8) and (25.9) together, the probability of a derivation f is calculated by multiplying together the probabilities of the fragments f_i which are composed together to form that fragment; this is analogous to how derivations are computed in Tree-DOP: there, we just have tree fragments, whereas in LFG-DOP, we have tree fragments together with their associated f-structure fragments.

In Tree-DOP, apart from the *Root* and *Frontier* operations, there are no other well-formedness checks. LFG, however, has a number of grammaticality conditions, some of which – the Completeness check at least – cannot be evaluated during the stochastic process. Accordingly, probabilities for valid representations can only be defined by sampling *post hoc* from the set of representations which are output from the stochastic process. The probability of sampling a valid representation is (25.10) (cf. example (12) in Bod & Kaplan 1998: 148):

$$P(R \mid \ R \text{ is valid}) = \frac{P(R)}{\sum_{R' \text{ is valid}} P(R')} \tag{25.10}$$

Bod & Kaplan (1998) note that (25.10) assigns probabilities to valid representations whether or not the stochastic process guarantees validity. The valid representations for a particular utterance u are obtained by a further sampling step, with their probabilities given by (25.11) (cf. example (13) in Bod & Kaplan 1998: 148):

$$P(R \mid \ R \text{ is valid and yields } u) = \frac{P(R)}{\sum_{R' \text{ is valid and yields } u} P(R')} \tag{25.11}$$

Comparing (25.10–25.11) with the equivalent formula for calculating the probability of a particular analysis for a Tree-DOP representation, Way (2001) observes that the LFG-DOP formulae contain references to *valid* structures. In Tree-DOP, apart from the root-matching criterion, there are no other validity conditions; in LFG-DOP, depending on the competition set chosen, there may be several.

Omitting the details for reasons of space, Bod & Kaplan (1998) give three different competition sets depending on the stage at which the LFG grammaticality checks are carried out, which affect the the **probability models** for LFG-DOP:

1. A straightforward extension of the Tree-DOP probability model, where the choice of a fragment depends only on its *Root* node (i.e. c-structure

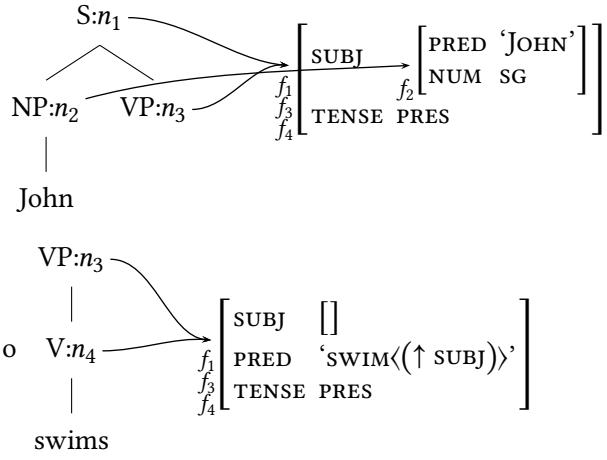
matching category) and not on the Uniqueness, Completeness or Coherence conditions of LFG, which are enforced off-line.

2. *Root* nodes must match, and f-structures must be unifiable if two LFG fragments are to be combined. This model takes the LFG Uniqueness condition (namely that each attribute has only one value) as well as the *Root* category into account. As the resultant fragments produced vary depending on the derivation followed, unifiability must be determined at each step in the process.

3. In addition to the previous two steps, the LFG Coherence check is enforced at each step, ensuring that each grammatical function (SUBJ, OBJ etc.) present in the f-structure is governed by a PRED. This means that in this model, we are dealing only with well-formed c-structures which correspond to coherent and consistent f-structures, i.e. structures which satisfy LFG's Uniqueness check, thereby permitting unification only where exactly appropriate. As we have noted already, the LFG Completeness check can only be enforced after all other validity sampling has taken place.

Let us now return to the sentence *John swims*, and show one possible derivation of the $\langle c, f \rangle$ pair in (1). A straightforward way of doing this would be to compose (via the 'o' operator in (12)) the $\langle c, f \rangle$ fragment in (3) with the leftmost fragment in (7), which we include in full in (12):

(12)



This is possible given that the VP node in the upper tree is vacant, so the lower VP tree can be substituted for this node. The respective f-structures are then unified to give the $\langle c, f \rangle$ fragment in (1). Throughout the derivation of this $\langle c, f \rangle$ pair, we have satisfied DOP's *Root* condition (leftmost substitution of 'like' categories only), as well as the Uniqueness, Completeness and Coherence grammaticality conditions of LFG. As a consequence, the resultant structures in (1) are valid. This is equivalent to using the third option given above for possible competition sets.

Of course there will be many other possible derivations which contribute to the overall probability of the sentence *John swims*. Note that if we enforce LFG's grammaticality checks on-line, leftmost substitution of non-*Discard* fragments reduces the size of the competition set for future iterations of the composition process. In (12), for instance, enforcing the Uniqueness condition on-line (models 2 or 3 above) prevents any fragment other than a singular intransitive VP from being substituted into the VP slot. In Tree-DOP, *any* VP could be substituted at this node.

3 LFG-DOT

In this section, we demonstrate that problems with the LFG-MT (Kaplan et al. 1989) and Data-Oriented Translation (DOT: Poutsma (2000)) models of translation can be solved by LFG-DOT.[6] As the LFG-DOT models proposed by Way (1999) and Way (2001) are based on LFG-DOP, they have the same advantages as shown in the previous section, albeit now for translation:

1. Being a probabilistic model, LFG-DOT can overcome problems encountered by LFG-MT which is based solely on LFG's constraints; and

2. By appealing to LFG's f-structure constraints, LFG-DOT can overcome problems encountered by DOT which is based solely on trees.

3.1 LFG-MT

A translation model based on LFG was first presented in Kaplan et al. (1989). This original model introduces the τ-correspondence as a mapping between source

[6]Hearne (2005) demonstrates that reasonably large-scale models can be built with DOT that considerably outperform SMT. Bod (2007) contains results which demonstrate similar improvements over SMT, but for *really* large-scale models at the time. Given the massive time and space constraints involved in processing DOP models, it is noteworthy that Bod was able to build DOT models trained on more than 750K sentence-pairs of German-English Europarl data (Koehn 2005).

and target f-structures. For *swim*, we would need a transfer lexicon entry such as (13) for translation between English and French:

(13) *swim*:
($\tau\uparrow$ PRED) = nager
($\tau\uparrow$ SUBJ) = τ($\uparrow$ SUBJ)

Being a straightforward translation example, this entry demonstrates two things: (i) that the translation of the verb *swim* is *nager*, and (ii) that the translation of the subject of *swim* is the subject of *nager*.

This model is very elegant, and allows for some difficult translation problems to be handled by the LFG-MT formalism. For example, verbs with different semantic forms can be handled relatively straightforwardly. Assume the translation case in (14):

(14) The student answers the question ⟷ L'étudiant répond à la question.

This case can be dealt with as in (15):

(15) *answer*:
($\tau\uparrow$ PRED) = répondre
($\tau\uparrow$ SUBJ) = τ($\uparrow$ SUBJ)
($\tau\uparrow$ OBL OBJ) = τ($\uparrow$ OBJ)

This states that *répondre* is the corresponding French predicate of *answer*, that the translation of the SUBJ is straightforward, and that the translation of the OBJ of *answer* is the OBL OBJ of *répondre*.

The LFG-MT model of Kaplan et al. (1989) can also deal correctly with the *like–plaire* relation-changing case, as (16) demonstrates:

(16) *like*:
($\tau\uparrow$ PRED) = plaire
($\tau\uparrow$ OBL$_)$ = τ($\uparrow$ SUBJ)
($\tau\uparrow$ SUBJ) = τ($\uparrow$ OBJ)

That is, the subject of *like* is translated as the oblique argument of *plaire*, while the object of *like* is translated as the subject of *plaire*.

However, a line of work showed that while the τ-equations of Kaplan et al. (1989) are by and large able to link exactly those source–target elements which are translations of each other, there are a number of cases where this machinery is unable to cope with a set of translation cases, in particular embedded

headswitching examples and the correct translation of adjuncts (cf. Arnold et al. 1990; Sadler & Thompson 1991; Way 2001).[7]

It is worth noting that an updated version of LFG-MT was described in Kaplan & Wedekind (1993) which used the concept of *Restriction* to try to overcome some of the problems in mapping between flat syntactic f-structures to hierarchical semantic ones. However, as well as receiving criticism from a monolingual perspective (cf. Butt (1994) and complex predicates in Urdu), Way (2001) demonstrates this new approach failed to ensure that only the correct translations ensued; rather, it was left to a human expert to select the correct translation from a set of alternatives, many of which were incorrect. Despite being an improvement on the original model of Kaplan et al. (1989), it is still open to criticism as a general model of translation.

Another solution proposed around this time involved using linear logic (van Genabith et al. 1998), but this involved adding massive redundancy in the transfer lexicon, cf. Way (2001: 92–96).

Note too that work continued on using LFG as a basis for MT after LFG-DOT was introduced. One such model was that of Riezler & Maxwell (2006). Note that their paper is not a comparison of LFG-MT, but rather with SMT (Koehn et al. 2003). Note that they add a 'fragment grammar' which "allows sentences that are outside the scope of the standard grammar to be parsed as well-formed chunks" (p.251), but they do not compare this with the bag of *Discard*-generated fragment-pairs in LFG-DOT. This work is extended by Graham & van Genabith (2012), who incorporate a deep syntax language model directly into the decoder, as opposed to using it *post hoc* to improve the grammaticality of the target translations. Note that neither approach shows how their models handle any of the traditional 'hard' translation cases. For the approach of Riezler & Maxwell (2006), being based on transfer rules – albeit automatically extracted ones – it will surely fail in similar ways to LFG-MT. As to the model of Graham & van Genabith (2012), and approaches based on SMT in general, it is doubtful whether the system designers can answer the question how such translational phenomena are handled, as SMT does not work in this way. Of course, test sets can be designed which include such 'hard' cases, and the translation output inspected, but SMT systems

[7] To give the reader some insight into the first-mentioned issue without having to consult the primary literature, LFG-MT can cope with 'straightforward' headswitching examples like *The baby just fell* ⟷ *Le bébé vient de tomber*. However, when such examples appear in embedded clauses, as in *I think that the baby just fell* ⟷ *Je pense que le bébé vient de tomber*, *ad hoc* solutions are required to avoid target f-structures being doubly rooted, i.e. two τ-equations result in inconsistent solutions where one piece of f-structure is required to simultaneously fill two different slots.

by their very nature are far less inspectable than systems which include syntactic constraints, so even if such sentences were translated correctly, it would be hard to know why exactly. Of course, this problem is worse again with today's state-of-the-art neural models; despite the improved quality that can be derived, our knowledge as to what is going on internally inside the systems is less than it's ever been.

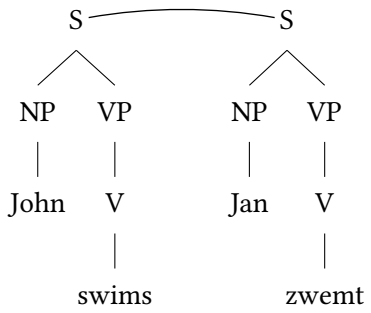
3.2 Data-Oriented Translation

Poutsma (2000) produced two models of tree-based translation, DOT1 and DOT2. These models were formulated along the same lines as DOP and LFG-DOP, with definitions of the representations to be used, how these were to be decomposed, recomposed, and a probability model.

In DOT, the latter determined the likelihood of a target translation given a source string. The representations used were PS trees, decomposition described how to extract well-formed subtree-pairs from these representations,[8] and the composition operator used was leftmost substitution (to ensure unique derivations) of matching *Root* labels.

We illustrate a linked translation pair in DOT in (17) for the the sentence pair ⟨ *John swims, Jan zwemt* ⟩:[9]

(17)

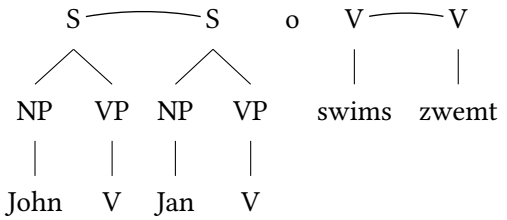


If we assume that the sentential fragment in (17) is unseen in our DOT treebank, one derivation of the translation *Jan zwemt* given the source sentence *John swims* might be (18):

[8] In his thesis, Way 2001 introduces the label γ to refer to the function that links DOT source and target subtree fragments. See Section 3.3.2 for models which use the γ function in LFG-DOT, and Poutsma 2000: Sect. 2.1 for a description of how linked subtrees like the V-labelled fragments in (18) are extracted from tree pairs such as (17).

[9] Here we 'translate' names to indicate that the translation process has been successful, as opposed to merely passing over a source word as untranslated – an out-of-vocabulary item – into the target side.

(18)



Way (1999) showed that the DOT1 model could not always explicitly relate parts of the source-language structure to the corresponding, correct parts in the target structure, so fails to translate correctly where source and target strings differ with respect to word order (e.g. the *like* ⇔ *plaire* relation changing case – which LFG-MT can handle, cf. (16) – plus many more 'hard' translation cases described in Way et al. (1997)).

DOT2 was developed as a consequence of these failings, and improves over DOT1 by not restricting the composition operation to left-most substitution on *both* sides. With that change, DOT2 manages to overcome cases of word-order difference by and large. However, Way (2001) notes that:

> "this is compromised by a lesser amount of compositionality in the translation process. Given the small number of fragments playing a role in the derivation of some translations involving complex phenomena, almost the exact linked sentence pair may need to be present in order for a translation to be possible. Furthermore, any such translations produced have extremely small probabilities with respect to the corpus. Finally, of course, translation systems which are based purely on PS trees will ultimately not be able to handle certain linguistic phenomena." (Way 2001: 190)

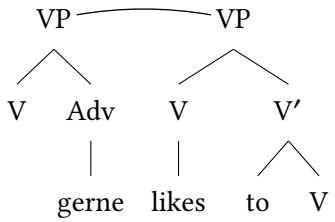
To illustrate the 'limited compositionality' problem in DOT2, Way (2001) appeals to the translation pair in (19):[10]

(19) DE: Johannes schwimmt gerne ⇔ EN: John likes to swim.

Essentially, the VPs cannot be broken down further; *schwimmt* and *swim* are not translationally equivalent – one is inflected and the other is in the infinitive form – so in their source–target tree pairs, links cannot be drawn between the fragment-pair in (20), as we might otherwise wish to do, in order to describe the basic translation relations in (19):

[10] Given that other similar cases exist, e.g. DE: Josef läuft zufällig ⇔ EN: Joseph happens to run, the redundancy in the DOT2 approach really shows itself to be problematic when such cases are combined, as in strings such as *John likes to happen to swim* (i.e. John likes to swim by chance, rather than planning ahead), and *John happens to like to swim.*

(20)



Accordingly, while it is possible for DOT2 to cope with such examples in contrast to DOT1, which couldn't handle them at all, the exact VPs (*likes to swim*, here) have to exist *a priori* in the treebank. This is because these linked VP pairs are handled non-compositionally in DOT2 between German and English, but the monolingual VPs are treated compositionally in DOP. As can be seen, DOT2 approximates to a translation dictionary for such cases – as *likes to* can be followed by pretty much any verb in English, and *gerne* can modify pretty much any verb in German – which is clearly impractical, and so can be disregarded as a general model of translation.

3.3 Combining DOT and LFG-MT: the best of both worlds

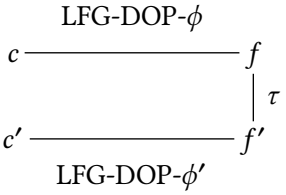
In his thesis, Way (2001) provides four LFG-DOT models which solve all these 'hard' cases:

1. Model 1: Translation via τ
2. Model 2: Translation via τ and γ
3. Model 3: Translation via γ with Monolingual Filtering
4. Translation via γ and 'Extended Transfer'

3.3.1 LFG-DOT1

Way (2001) describes this as a simple linear model, as in (21):

(21)



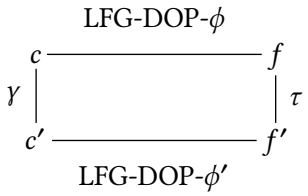
The different components needed are:

- a source language LFG-DOP model;
- the τ mapping;
- a target language LFG-DOP model.

Way (2001: 193) notes that "LFG-DOT1 contains two monolingual LFG-DOP language models … [so] *Discard* can be run on both source and target sides. This means that LFG-DOT1 can cope with ill-formed or previously uncovered input which LFG-MT would not be able to handle at all". Despite this advantage, LFG-DOT1 unsurprisingly suffers from the same problems as LFG-MT, as its translation function is described by the same operator τ.

3.3.2 LFG-DOT2

Given that τ is an insufficient operator to define all translation problems (cf. fn. 7, for example), Way (2001) describes the translation relation using both the γ and τ functions in his LFG-DOT2 model, summarised in (22):

(22)


This is clearly a more complex model than LFG-DOT1, necessitating:

- a source language LFG-DOP model;
- the γ mapping (i.e. the DOT2 model of translation, cf. fn. 8);
- a target language LFG-DOP model;
- a probabilistic transfer component.

Way (2001) provides a number of ways in which the γ and τ functions might co-operate in his LFG-DOT2 model. He notes that using LFG-DOP as the source and target language models overcomes the shortcomings of both Tree-DOP and LFG, and that including τ allows certain 'hard' cases (like relation-changing) to be handled correctly, unlike the DOT1 model.

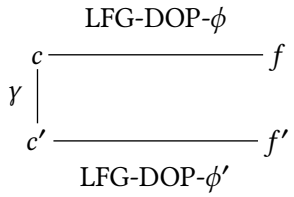
Furthermore, Way (2001) notes that LFG-DOT2 is more robust than LFG-MT, in that *Discard* can produce generalized fragments which may be able to deal with input for which LFG-MT cannot offer any translation.

Ultimately, as the τ mapping cannot always produce the desired translation, Way jettisons this function in his LFG-DOT3 and LFG-DOT4 models, to which we turn next.

3.3.3 LFG-DOT3

LFG-DOT3 relies solely on γ to express the translation relation. The architecture of LFG-DOT3 is shown in (23):

(23)



Way (2001) demonstrates that contrary to other models described here, embedded headswitching cases in LFG-DOT3 are handled in the same manner as non-embedded headswitching cases, exactly as required (cf. fn. 7). He also shows that LFG-DOT3 can cope with certain cases of combinations of exceptional phenomena which prove problematic for other formalisms. However, like DOT2 (cf. Section 3.2), LFG-DOT3 also suffers from the problem of limited compositionality.

3.3.4 LFG-DOT4

To overcome this problem, Way (2001) uses a restricted form of *Discard* in an 'extended transfer' phase in LFG-DOT4 to generalize the translation relation appropriately. Essentially, in LFG-DOP (and consequently LFG-DOT), fragments generated by *Discard* occupy an unjustifiably large proportion of the probability space. Accordingly, Way (1999) proposes to split fragments into two bags: those generated by *Root* and *Frontier*, and those generated by *Discard*. In LFG-DOT4, Way (2001) allocates a small amount of the probability space to lemmatized translation pairs produced by a second application of *Discard*.[11] To revisit

[11] Another way of mitigating this problem is suggested by Way (2001: 112), namely to adopt the approach of Zaenen & Kaplan (1995), which cuts down on the possible number of LFG-DOP fragments compared to the description of LFG in Kaplan & Bresnan (1982). In Zaenen & Kaplan (1995), lexical heads are ϕ-linked only to semantic forms and not to their enclosing f-structures, while other primitive feature values remain unlinked.

the problematic example in (19), if *Discard* is used to relax the TENSE constraint, then the V nodes in (20) can be linked; they couldn't before as the V in German was a finite verb, while the V in English was an infinitive. Accordingly, Way (2001: 190) suggests that "this model describes the translation relation exactly as required, and furthermore overcomes the problems of LFG-MT ... and DOT models of translation". See Table 1 for a summary of the comparative advantages and disadvantages of each of the models covered in this chapter.[12]

Table 1: A comparison of the advantages and disadvantages of the MT models described in this work

Model	Ill-formed input	Word order	Embedded headswitching	All 'hard' cases	Avoids limited compositionality
LFG-MT	N	Y	N	N	N
DOT1	Y	N	N	N	N
DOT2	Y	Y	Y	N	N
LFG-DOT1	Y	Y	N	N	N
LFG-DOT2	Y	Y	Y	N	N
LFG-DOT3	Y	Y	Y	Y	N
LFG-DOT4	Y	Y	Y	Y	Y

4 Automatic derivation of f-structures from treebanks

In this section we consider how the resources needed for large-scale LFG-DOP and LFG-DOT models can be generated. We also explain the two different ways in which f-structures can be derived from a tree.

4.1 Towards large-scale resources for LFG-DOP and LFG-DOT

LFG-DOP needs large collections of monolingual annotated data (treebanks) in order to parse monolingual input, and LFG-DOT needs large collections of bilingual annotated data. At the time LFG-DOP and LFG-DOT were being developed, no such large f-structure annotated data existed. Constituency treebanks had

[12] See Way (2003) for more details on these models, and Hearne (2005) for an alternative LFG-DOT model based on LFG-DOT3 but which incorporates a different probability model and fragmentation procedure.

been available for several years, and large-scale hand-crafted LFG grammars were available only for a few languages. However, neither could provide the input needed to support the training of LFG-DOP or LFG-DOT models. Constituency treebanks alone could not provide the linguistic detail needed, and hand-crafted grammars were unable to select the most likely parse from a (sometimes) large number of possible solutions.

To address these shortcomings, van Genabith, Way, et al. (1999b) and van Genabith, Sadler, et al. (1999) proposed initial methods to automatically derive the LFG-treebank resources required to support training LFG-DOP and LFG-DOT models, although this was not the main driving force behind this work.

Initially, the work conducted produced grammars and lexicons for English, which seeded high-performing probabilistic parsers (see Section 5). Later, related methods were used to extract similar resources for a range of other languages (cf. Section 6). Once the general approach had been validated for different languages and treebanks, it is possible to sketch a research project which could generate the resources needed for large-scale LFG-DOP and LFG-DOT experimentation.

Taking a large-scale parallel corpus such as Europarl (Koehn 2005), we would need to:

1. Parse source and target sides to generate c-structure trees for the two languages;
2. Run the f-structure annotation algorithm(s) over each side;
3. Apply the *Root* and *Frontier* operations to extract the separate bags of fragments.

After step 2, we have $\langle c, f \rangle$ pairs of structure for all sentences on both sides of the corpus, so we can build LFG-DOP models for the individual source and target sides by running *Root* and *Frontier* operations on each side, and start producing $\langle c, f \rangle$ pairs for new monolingual input. To generate resources for the better of the four models, LFG-DOT4, we need to align each source tree generated in step 1 with each target tree generated in the same step. Fortunately, Europarl contains information regarding which sentences in one language map to which sentences in another, so we can now apply *Root* and *Frontier* operations on both sides to extract the separate bags of fragments that are needed, and start translating new input strings. This experiment remains for future work.

4.2 Direct Transformation vs. Indirect Annotation

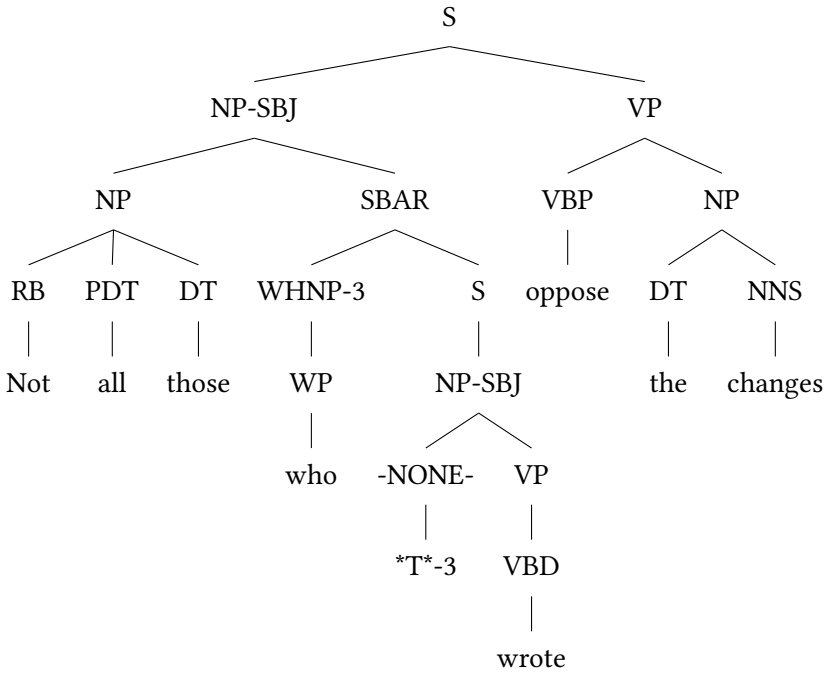
The initial approaches of van Genabith, Way, et al. (1999b) and van Genabith, Sadler, et al. (1999) focused on deriving f-structure annotations from PS trees. The

intuition was that there were already reasonably reliable tools for automatically producing a tree from an input sentence, and so it would be easier to scale a tree-annotation plus f-structure derivation approach, compared to automatically deriving c- and f-structure simultaneously from raw input.

There are two ways to derive an f-structure from a tree: **direct** transformation or **indirect** annotation. The direct method recursively and destructively transforms a treebank tree into an f-structure. The indirect method only ever adds information: it annotates the treebank tree with f-structure annotations (equations). These annotations are then collected and passed to a constraint solver which resolves the equations and, if the equations are consistent, outputs an f-structure.

Examples (24)–(26) illustrate the indirect method: all nodes in the tree in (24) are annotated with equations in (25), which are collected and resolved into an f-structure in (26).

(24)



(25)
```
(S
    (NP-SBJ[up-subj=down]
        (NP[up=down]
            (RB[down-elem=up:adjunct] Not[up-pred='not'])
            (PDT[up-spec:det=down] all[up-pred='all'])
            (DT[up=down] those[up-pred='those'])
        )
        (SBAR[up-relmod=down]
            (WHNP-3[up-topicrel=down,up-topicrel:index=3]
                (WP[up=down] who[up-pred=pro,up-pron_form='who'])
            )
            (S[up=down]
                (NP-SBJ[up-subj=down,up-subj=up:topicrel]
                    (-NONE- *T*-3)
                )
                (VP[up=down]
                    (VBD[up=down] wrote[up-pred='write',up-tense=past])
                )
            )
        )
    )
    (VP[up=down]
        (VBP[up=down] oppose[up-pred='oppose',up-tense=pres])
        (NP[up-obj=down]
            (DT[up-spec:det=down] the[up-pred='the'])
            (NNS[up=down] changes[up-pred='change', up-num=pl,up-pers=3])
        )
    )
    (. .)
)
```

(26)
```
subj : adjunct : 1 : pred : not
       spec : det : pred : all
       pred : those
       relmod : topicrel : index : 3
                           pred : pro
                           pron_form : who
                subj : index : 3
                       pred : pro
                       pron_form : who
                pred : write
                tense : past
pred : oppose
tense : pres
obj : spec : det : pred : the
      pred : change
      num : pl
      pers : 3
```

The earliest approach to automatically identifying functional grammatical categories such as SUBJ, OBJ, etc in PS trees is probably that of Lappin et al. (1989). Nodes in trees are linked to their corresponding grammatical functions. Their motivation was to generate a set of grammatical function-based transfer rules as part of an MT project.

A regular expression-based, indirect automatic annotation method is described in Sadler et al. (2000). This involves extracting a context-free PS grammar (CFG) from a treebank fragment. F-structure annotation principles are stated in terms of regular expressions matching CFG rules. By applying regular expression-based annotation principles to the rules that are extracted, and using these annotated rules to re-match the original trees, f-structures can be generated for these trees. The number of annotation principles is appreciably smaller than the number of extracted CFG rule types since the regular expression-based annotation principles capture linguistic generalisations.

The flat, set-based tree description rewriting method of automatically annotating trees with f-structure descriptions developed by Frank (2000) can be seen as a generalisation of the regular expression-based technique of Sadler et al. (2000). Here the idea is that each tree is translated into a flat description using terms from a tree description language (e.g. *lex, arc, phi* etc.). Annotation principles are then defined in terms of rules employing a rewriting system originally developed for transfer-based MT architectures (Kay et al. 1994). In certain circumstances, the principles can be applied order-independently, or in a particular cascading order. One of the advantages of this method is that tree fragments of arbitrary depth can be considered, whereas in the regular expression-based method, tree depth is limited to 1 (i.e. CFG rules).

The earlier approaches were limited in scale to corpora in the order of hundreds of trees. In Cahill et al. (2002a), a first version of a large-scale indirect annotation algorithm was described. This algorithm was scaled to a corpus containing tens of thousands of trees. The algorithm recursively traverses a PS tree and annotates f-structure information on each node. McCarthy (2003) and Burke (2006) continued to expand this algorithm in terms of linguistic coverage. The algorithm itself is modular and separates the linguistic data from the traversal algorithm. There are two stages to the algorithm: (i) "proto"-f-structures are generated which contain unresolved long-distance dependencies (LDDs); and (ii) trace information encoded in the treebank is used to correctly link moved constituents to where they should be interpreted semantically. Given a PS tree with f-structure-annotated nodes, a constraint solver based on the one described in Gazdar & Mellish (1989) was used to produce the final f-structure representation

for the tree. This body of work yielded the first large-scale algorithm for converting a treebank into a corpus of f-structures. This was a prerequisite for the parsing work that built on this corpus as described in Section 5.

Similar efforts to automatically acquire wide-coverage grammars for TAG (Xia 1999), HPSG (Miyao et al. 2003), and CCG (Hockenmaier & Steedman 2002) appeared around the same time as the work on LFG.

5 Probabilistic parsing & lexicon induction using LFG

With the availability of large-scale f-structure-annotated treebanks, it was now possible to train probabilistic LFG parsers.

The initial parsing experiments of Cahill et al. (2002b) were conducted on the Penn Treebank (Marcus et al. 1994). Two main approaches were compared:

1. Parse with a standard CFG parser and then automatically annotate the resulting tree (*pipeline* architecture)

2. Automatically annotate the nodes in the trees of a large corpus with f-structure information and train a probabilistic parser on it (*integrated* architecture)

Both approaches yielded c-structures whose node labels included f-structure annotations. These f-structure annotations were then collected and resolved to generate a final f-structure. Initial parsers generated what were called "proto"-f-structures which did not include any LDD resolution. It should be noted that since these techniques were probabilistic, a set of n-best trees (and therefore f-structures) could also straightforwardly be produced. This was not possible with hand-crafted grammars which output all possible f-structure solutions for a given sentence without any way to sort them. Riezler et al. (2002) showed that it was possible to *post hoc* rank the output of such a parser, however.

In Cahill (2004) and Cahill et al. (2004), additional functionality was added to the original algorithm to allow for LDD resolution. This yielded more complete f-structures. There were two main components to the algorithm: (i) a set of possible functional uncertainty paths, and (ii) a subcategorisation lexicon.

In order to obtain the set of possible functional uncertainty paths, all observed paths between co-indexed material were extracted from the f-structures automatically derived from the Penn Treebank. These paths were associated with probabilities. O'Donovan et al. (2004) and O'Donovan (2006) describe an approach for automatically acquiring a large-scale subcategorisation lexicon from the Penn

Treebank. This relies on the intuition that if the original conversion of the treebank into f-structures is of high enough quality, then the lexical entries for all predicates can be reverse-engineered (van Genabith, Way, et al. 1999a). Frames are not predefined, yet the frames that are automatically acquired fully reflect LDDs in the source data-structures, discriminate between active and passive frames, and conditional probabilities are associated with each frame.

Given a set of semantic forms s with probabilities $P(s|l)$ (where l is a lemma), a set of paths p with $P(p|t)$ (where t is either TOPIC, TOPICREL or FOCUS) and an f-structure f, the core of the algorithm to resolve LDDs recursively traverses f to identify the most likely location of co-indexed material.

Evaluation of the f-structures produced by both parsing approaches was carried out against several corpora over time: the DCU-105 corpus (Cahill et al. 2002a), the automatically converted Section 23 of the Penn Treebank, the PARC 700 corpus (King et al. 2003) and the CBS 500 (Carroll et al. 1998). F-structures were converted into dependency triple format and compared to the gold-standard triples to give results in terms of precision, recall and f-score. Results demonstrated state-of-the-art results compared to other 'deep' parsers available at the time. Cahill, Burke, O'Donovan, Riezler, et al. (2008) summarize a large set of parser comparisons, and show that the f-structures produced by the automatic processes described above were able to outperform two hand-crafted parsers: RASP (Carroll & Briscoe 2004) and the the English ParGram LFG run on XLE (Riezler et al. 2002). Rimell et al. (2009) conduct a comparison of several "deep" parsers on a specialized corpus of sentences containing only LDDs. They find that the HPSG and CCG parsers perform better than the DCU LFG parser on this set of difficult sentences.

6 Multilingual probabilistic LFG induction

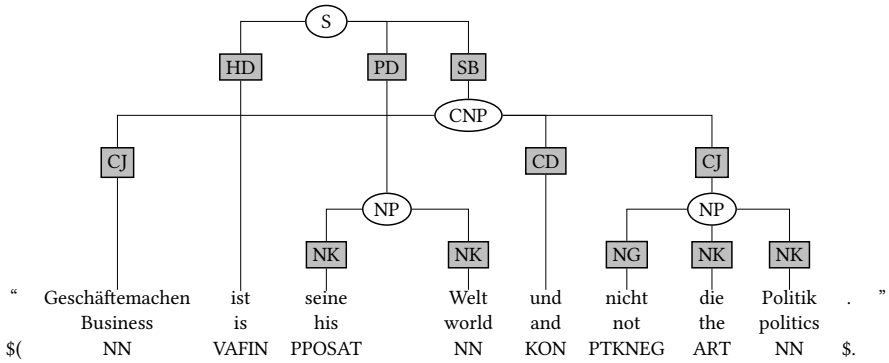
The approach developed for English was language-independent. Given a large enough and detailed treebank, one could theoretically follow the same framework to generate parsers and lexicons for other languages. Indeed, given that comparable treebank existed for some languages other than English, a large body of work ensued in this direction.

Cahill et al. (2003) first attempted this for German using the TiGer Treebank (Brants et al. 2002). This treebank differs from the Penn Treebank in that it encodes parses in terms of labeled graphs that allow crossing edges. In Cahill et al. (2003), the graphs are first converted to trees similar to those found in the Penn Treebank with trace information added to account for moved constituents.

A set of rules was then developed that automatically assigned f-structure equations to the nodes in the trees, and the same techniques described in Cahill et al. (2002b) were used to automatically acquire the first large-scale probabilistic LFG for German.

We provide an example graph from the TiGer treebank in (27) for the German sentence "Geschäftemachen ist seine Welt und nicht die Politik" ("Business is his world, not politics").

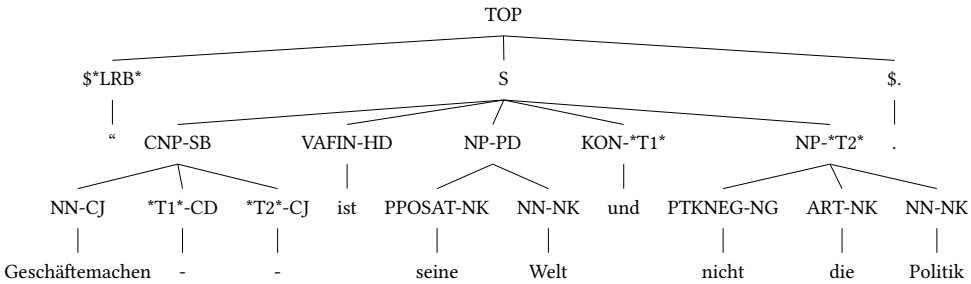
(27)



'Business is his world, not politics.'

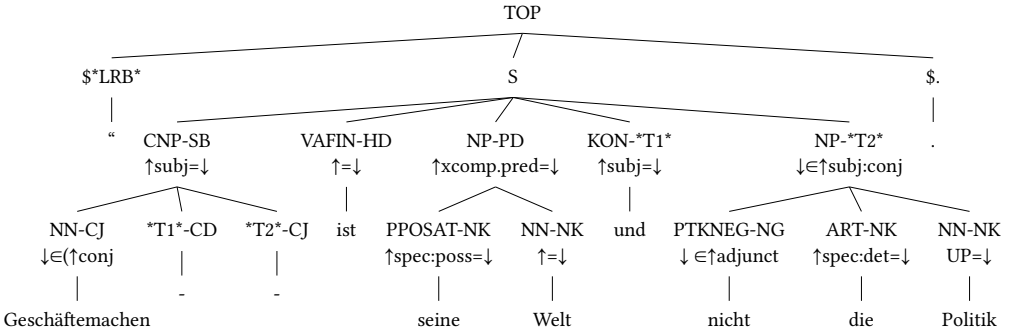
In (28), the graph in (27) is first automatically converted into a PS tree with traces and coindexation to indicate linked elements, analogous to how this kind of information is encoded in the English Penn-II treebank.

(28)



In (29), the tree in (28) is then annotated with f-structure equations. The annotation algorithm relies heavily on the functional component of the tree node labels (e.g. that SB indicates a SUBJECT).

(29)



Finally, the equations in (25) are collected and passed through a constraint solver to generate the f-structure in (30), using the same procedure as for English.

(30)
$$\begin{bmatrix} \textsc{xcomp.pred} & \begin{bmatrix} \textsc{spec} & \begin{bmatrix} \textsc{poss} & \begin{bmatrix} \textsc{pred} & \textsc{pro} \end{bmatrix} \end{bmatrix} \\ \textsc{pred} & \textsc{Welt} \end{bmatrix} \\ \textsc{subj} & \begin{bmatrix} \textsc{conj} & \left\{ \begin{bmatrix} \textsc{pred} & \textsc{Geschäftemachen} \end{bmatrix}, \begin{bmatrix} \textsc{pred} & \textsc{Politik} \\ \textsc{spec} & \begin{bmatrix} \textsc{det} & \begin{bmatrix} \textsc{pred} & \textsc{die} \end{bmatrix} \end{bmatrix} \\ \textsc{adjunct} & \left\{ \begin{bmatrix} \textsc{pred} & \textsc{nicht} \end{bmatrix} \right\} \end{bmatrix} \right\} \\ \textsc{coord-form} & \textsc{und} \end{bmatrix} \\ \textsc{pred} & \textsc{ist} \end{bmatrix}$$

Rehbein & van Genabith (2009) continued this work and explored the effect of the design of the treebank on the success of the technique. They compared extracting a probabilistic LFG from both TiGer and TüBa-D/Z (Telljohann et al. 2006) and found (1) that automatically inducing linguistic resources from (semi-) free word order languages such as German is much harder than for more configurational languages like English, and (2) that the the treebank encoding can have a significant effect on the success of the automatic f-structure annotation approach. Rehbein & van Genabith (2009) found that the encoding of linguistic structures in the TiGer treebank was better suited for automatic induction of LFG resources, because it was more difficult to automatically learn the grammatical function relations as they were encoded in the TüBa-D/Z.

For Chinese, Burke, Lam, et al. (2004) first applied the approach to the Penn Chinese Treebank (Xue et al. 2002). We provide in (31) an example tree from this treebank.

(31) (IP-HLN
(NP-PN-SBJ
(NR 江泽民)
(NR 李鹏))
(VP
(VV 电唁)
(NP-OBJ
(NP-PN
(NR 尼克松))
(NP
(NN 逝世)))))
“江泽民李鹏电唁尼克松逝世”
‘Jiang Zemin and Li Peng condoled the bereavement of Nixon by a telegram.’

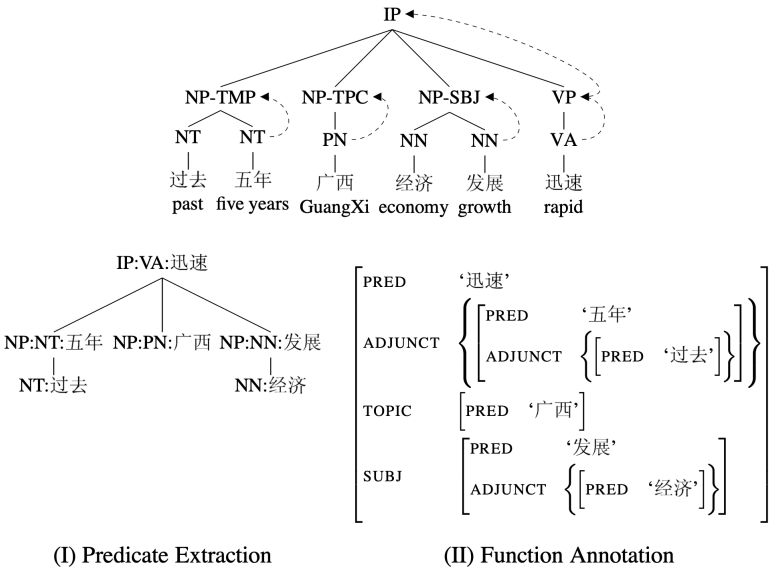
Each node in the tree in (31) is then annotated with f-structure equations, and the f-structure in (32) is derived.

(32)
```
subj :      coord_form : null
            coord :   1 :  pred : '江泽民'
                           pers : 3
                           noun_type : proper
                           gloss : ‘Jiang_Zemin’
                      2 :  pred : '李鹏'
                           pers : 3
                           noun_type : proper
                           gloss : ‘Li_Peng’
pred : '电唁'
gloss : condole_by_a_telegram
obj :       adjunct :  3 :  pred : '尼克松'
                           pers : 3
                           noun_type : proper
                           gloss : ‘Nixon’
            pred : '逝世'
            pers : 3
            noun_type : common
            gloss : ‘bereavement’
```
“江泽民李鹏电唁尼克松逝世”
“Jiang Zemin and Li Peng condoled the bereavement of Nixon by a telegram.”

Guo et al. (2007) and Guo (2009) extended this work in terms of coverage, robustness, quality and fine-grainedness of the resulting LFG resources. They propose a more general two-stage annotation architecture, avoiding some of the limitations of the PS annotation-based method. They argue that this approach may be more suitable for less configurational languages. This algorithm works by transducing the tree into an f-structure by means of an intermediate dependency structure.

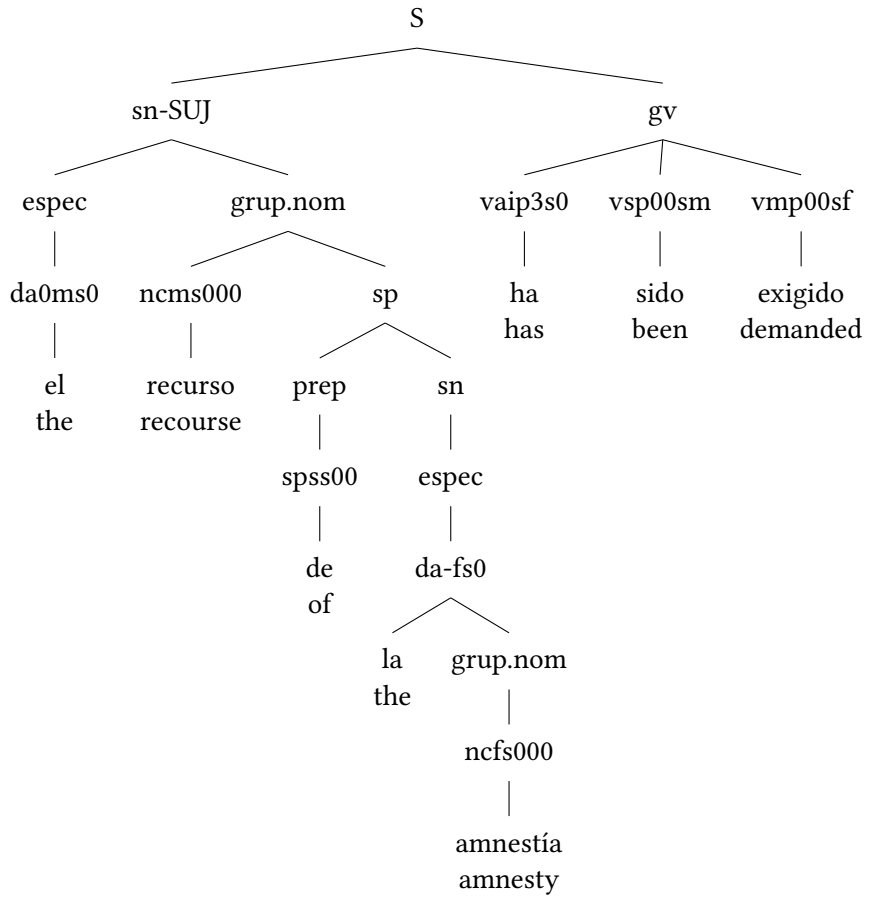
In (33), we show an example where predicate information is first extracted from the tree, and then a simpler set of function-based annotations converts the intermediate structure into an f-structure. The advantages of this approach are that it guarantees a single connected f-structure, as well as simplifying the process of taking LDDs into account.

(33)



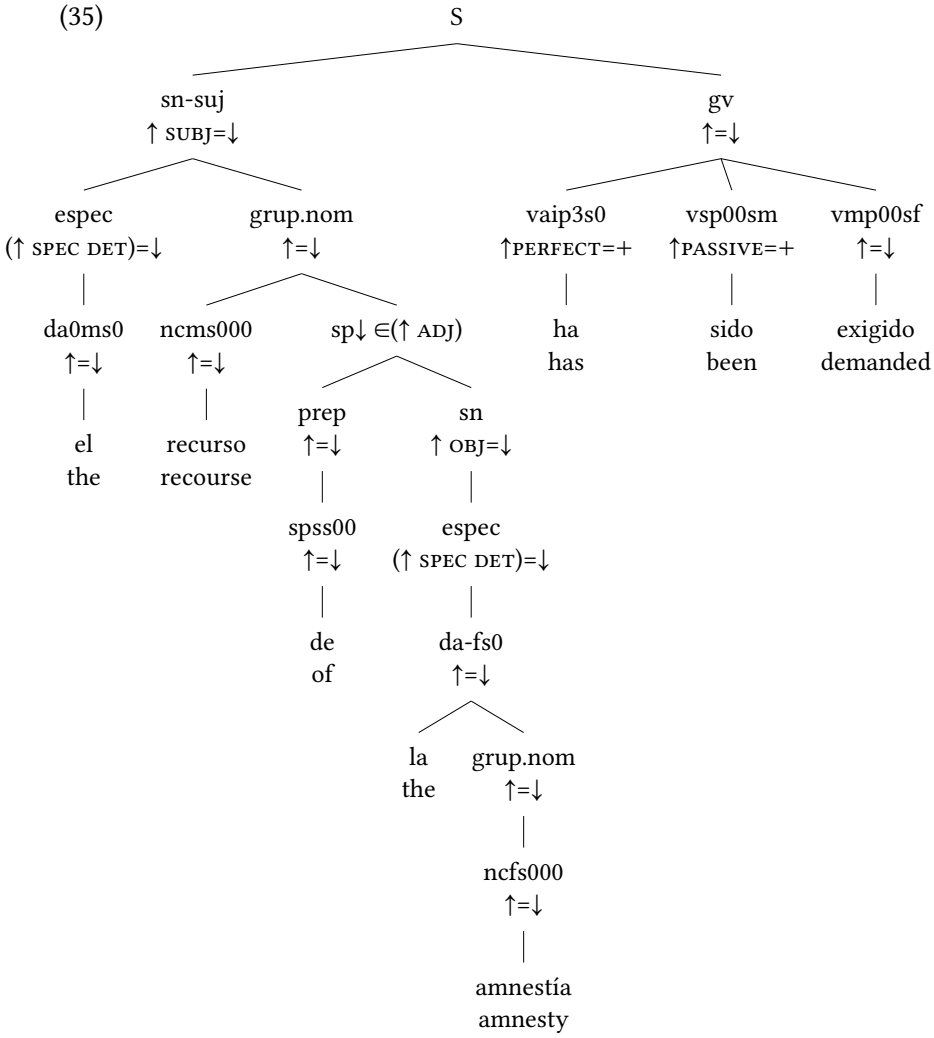
O'Donovan et al. (2005) proposed an adaptation of the original approach for Spanish using the CAST3LB treebank (Civit 2003). In (34), we provide an example from the CAST3LB treebank.

(34)



The tree in (34) is then annotated with equations, as illustrated in (35). The equations are then resolved into the f-structure in (36).

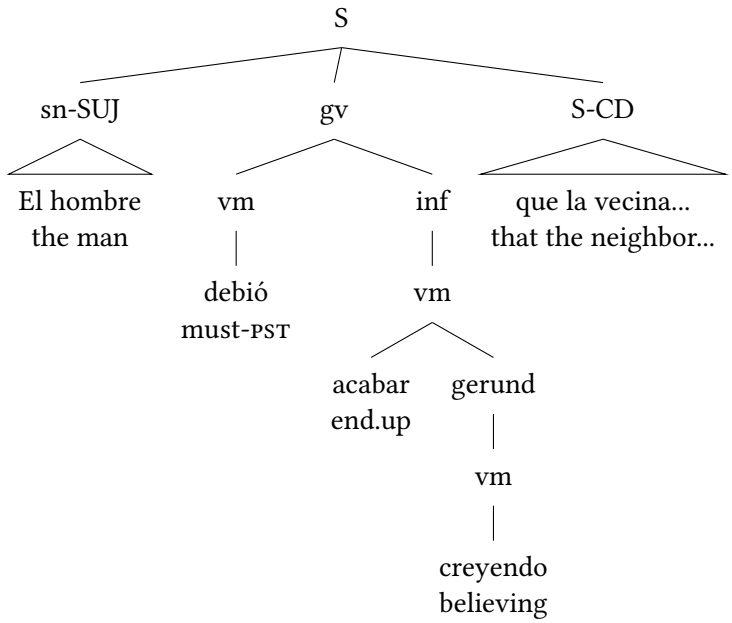
(35)


S
sn-suj
↑ SUBJ=↓
gv
↑=↓
espec
(↑ SPEC DET)=↓
grup.nom
↑=↓
vaip3s0
↑PERFECT=+
vsp00sm
↑PASSIVE=+
vmp00sf
↑=↓
da0ms0
↑=↓
ncms000
↑=↓
sp↓ ∈(↑ ADJ)
ha
has
sido
been
exigido
demanded
el
the
recurso
recourse
prep
↑=↓
sn
↑ OBJ=↓
spss00
↑=↓
espec
(↑ SPEC DET)=↓
de
of
da-fs0
↑=↓
la
the
grup.nom
↑=↓
ncfs000
↑=↓
amnestía
amnesty

(36)
$$\begin{bmatrix}
\text{PRED} & \text{'EXIGIR'} \\
\text{PERFECT} & + \\
\text{PASSIVE} & + \\
\text{TENSE} & \text{PRESENT} \\
\text{SUBJ} & \begin{bmatrix}
\text{SPEC} & \begin{bmatrix} \text{DET} & \begin{bmatrix} \text{PRED} & \text{'EL'} \\ \text{NUM} & \text{SING} \\ \text{GEND} & \text{MASC} \end{bmatrix} \end{bmatrix} \\
\text{PRED} & \text{'RECURSO'} \\
\text{NUM} & \text{SING} \\
\text{GEND} & \text{MASC} \\
\text{ADJ} & \left\{ \begin{bmatrix}
\text{PRED} & \text{'DE'} \\
\text{OBJ} & \begin{bmatrix}
\text{SPEC} & \begin{bmatrix} \text{DET} & \begin{bmatrix} \text{PRED} & \text{'EL'} \\ \text{NUM} & \text{SING} \\ \text{GEND} & \text{FEM} \end{bmatrix} \end{bmatrix} \\
\text{PRED} & \text{'AMNESTÍA'} \\
\text{NUM} & \text{SING} \\
\text{GEND} & \text{FEM}
\end{bmatrix}
\end{bmatrix} \right\}
\end{bmatrix}
\end{bmatrix}$$

This was extended in Chrupała & van Genabith (2006) where three main issues were addressed: (i) new constructions that had standard LFG analyses (e.g clitic doubling and null subjects); (ii) new constructions where no LFG analysis was available (e.g. periphrastic constructions in Spanish, see Figure 1); and (iii) limitations of the previous approach due to treebank- and language-specific assumptions which did not hold for Spanish and the CAST3LB treebank. Similar to what Guo et al. (2007) and Rehbein & van Genabith (2009) had found in their adaptations, the original approach assumed that the functional information could easily be derived from the tree configuration, but this proved not to be the case for many languages. Therefore, the functional tags in the parser output were critical for the success of these annotation algorithms. As a result, Chrupała & van Genabith (2006) outlined an improved method for tagging functions in parse trees, not only for Spanish, but for English, too. This was an important step in the development of a probablistic Spanish LFG parser based on the CAST3LB treebank.

In the case of French, no suitable treebank was immediately available. Therefore, Schluter & van Genabith (2007) first modified the Paris 7 Treebank (Abeillé et al. 2004), as this was the closest in format to what would be needed. A subset of the original treebank was transformed to yield a leaner, more coherent, treebank with several transformed structures, and new linguistic analyses. In Schluter & van Genabith (2008), it was shown that a probabilistic parser trained on the cleaner, modified treebank performed better than a parser trained on the



$$\begin{bmatrix}
\text{SUBJ} & \boxed{1}[\text{“EL HOMBRE”}] \\
\text{PRED} & \text{‘DEBER’} \\
\text{TENSE} & \text{PAST} \\
\text{LIGHT} & + \\
\text{XCOMP} & \begin{bmatrix}
\text{SUBJ} & \boxed{1} \\
\text{PRED} & \text{‘ACABAR’} \\
\text{LIGHT} & + \\
\text{XCOMP} & \begin{bmatrix}
\text{SUBJ} & \boxed{1} \\
\text{PRED} & \text{‘CREER’} \\
\text{LIGHT} & - \\
\text{COMP} & [\text{“QUE LA VECINA...”}]
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}$$

Figure 1: Treatment of periphrastic constructions outlined in Chrupała & van Genabith (2006)

much larger, but noisier, original treebank. In addition, Schluter & van Genabith (2008) and Schluter (2011) showed that the techniques for automatically acquiring LFG resources from treebanks could successfully be adapted to the French case. Thanks to a rich morphological and functional annotation in the treebank, the automatic annotation algorithm can rely on node labels rather than inferring functional labels via tree configurations. This leads to fewer incomplete f-structures, and fewer cases where LDDs have not been resolved.

Oya & van Genabith (2007) showed that the approach can also be adapted for Japanese using the Kyoto Text Corpus (Kurohashi & Nagao 1997). The Japanese corpus encodes syntactic units in addition to rich morphological information. The automatic annotation algorithm adds f-structure equations at the level of syntactic unit. Figure 2 shows how the f-structure equations are added to each syntactic unit of the sentence "Taro went to Seoul". In the case of Japanese LFG parsing, the key to successful parsing results was in zero pronoun identification.

Finally, Tounsi et al. (2009a) and Tounsi et al. (2009b) demonstrated that the approach was also possible for Arabic using the Penn Arabic Treebank (Maamouri & Bies 2004). The annotation algorithm was able to take advantage of rich morphological tags in the treebank to support the fact that Arabic is a morphologically rich language.

In most cases we observe that the original reliance on tree configurations to identify functional properties worked best for English. For the other languages, relying on functional information already in the original treebank, and then ensuring that the CFG parser also contains that information, yielded the most accurate f-structure parsers. Evaluation of LFG parsing for the other languages followed roughly the same procedure as for English, using a small manually annotated corpus of sentences from the treebank used to derive the algorithm and parser.

7 Related approaches to grammar induction

A natural evaluation of this approach to creating large-scale probabilistic LFG parsers is to compare large-scale grammars created manually using the XLE platform.

The method proposed in Riezler et al. (2002) provides a mechanism for ranking all possible solutions generated by the hand-crafted grammar, relying on the same kinds of treebank resources as the methods described above. Kaplan et al. (2004) show that the accuracy of the hand-crafted grammar is more accurate than the Collins (1999) parser (f-score of 77.6 vs 74.6), while only slightly slower

```
#S-ID:950101001-001
* 0 2D
太郎  たろう          * 名詞 人名 * *      (Taro Noun Person**)
が  が                * 助詞 格助詞 * *    (ga particle Case **)
F0:pred ='Taro',
F0:case='ga',
F2:subj=F0,
* 1 2D
ソウル  そうる        * 名詞 地名 * *      (souru "Seoul" * Noun Place**)
に                    * 助詞 格助詞 * *    (ni particle Case**)
F1:pred='Seoul',
F1:case='ni',
F2:obl=F1,
* 2 -1D
行った  いった  行く  動詞 * 子音動詞 過去形 (itta 'went' iku Verb * ConsonantStem pst)
F2:pred='iku',
F2:tns='pst',
F2:stmt='decl',
F2:style='plain'.
EOS
```

(a) The automatically annotated sentence

$$\begin{bmatrix} \text{subj} & \begin{bmatrix} \text{pred} & \text{'Taro'} \\ \text{case} & \text{ga} \end{bmatrix} \\ \text{obl} & \begin{bmatrix} \text{pred} & \text{'Seoul'} \\ \text{case} & \text{ni} \end{bmatrix} \\ \text{pred} & \text{'iku}\langle\text{subj, obl}\rangle\text{'} \\ \text{stmt} & \text{'decl'} \\ \text{style} & \text{'plain'} \\ \text{tense} & \text{pst} \end{bmatrix}$$

(b) The resulting f-structure

Figure 2: An example from the Kyoto Text Corpus: from syntactically annotated sentence to f-structure

(total 299 CPU seconds vs 200 CPU seconds to parse 560 sentences). The two approaches have the same goal: to provide a ranked list of LFG parses for a given input. The difference is in how this ranked list is derived, and how much manual effort is required. Furthermore, in Cahill, Maxwell, et al. (2008) it was shown that a simple pruning mechanism on the c-structure forests generated by the XLE parser could significantly reduce parsing time, while maintaining comparable accuracy.

8 Conclusion

This chapter has described methods based on LFG that permit accurate, robust, scalable, probabilistic LFG parsers and MT systems to be built from large collections of automatically annotated data. While this is commonplace nowadays, it was much less so 20–25 years ago.

LFG-DOP extends LFG by generalizing well-formed analyses to allow ill-formed and previously uncovered strings to be handled. LFG-DOT, a robust, hybrid model of translation based on LFG-DOP, was demonstrated to be able to solve 'hard' cases of translation that proved difficult for DOT and LFG-MT.

The range of work on automatic annotation of LFG grammars summarised here was an important step in ensuring scalability and robustness that is commonplace nowadays. Once large-scale treebanks could be generated via these techniques, competitive probabilistic parsers were built, and large-scale lexical resources were induced. However, most experiments carried out using LFG-DOP (and LFG-DOT) were relatively small-scale, but we sketch here a method for large-scale experimentation using the resources created via the techniques described in this paper.

As well as the important extension of the core LFG framework to account for probabilistic parsing, this seminal work also provided the foundations for the now commonplace task of large-scale deep linguistic LFG annotation. In sum, the work described in this chapter laid the foundations for multilingual annotation of treebanks, which in turn allowed competitive scalable parsing and MT models to be developed that are accepted as best practice today.

Acknowledgements

We thank the anonymous reviewers for their comments, which have helped to improve this paper. We are grateful to Mary Dalrymple for her input regarding Pirahã. The second author is supported by the ADAPT SFI Centre for Digital Content Technology, which is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and co-funded under the European Regional Development Fund.

References

Abeillé, Anne, François Toussenel & Martine Chéradame. 2004. *Corpus le monde: Annotations en constituants. Guide pour les correcteurs*. Tech. rep. Paris: Laboratoire de Linguistique Formelle, Université Paris Diderot.

Arnold, Doug, Louisa Sadler, Ian Crookston & Andy Way. 1990. LFG and translation. In *Proceedings of the Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*, 121–130. Austin.

Bod, Rens. 1992. A computational model of language performance: Data-Oriented Parsing. In *COLING '92: proceedings of the 14th international Conference on Computational Linguistics*, vol. 3, 855–859. Nantes. DOI: 10.3115/992383.992386.

Bod, Rens. 1998. *Beyond grammar: An experience-based theory of language*. Stanford, CA: CSLI Publications.

Bod, Rens. 2000. An empirical evaluation of LFG-DOP. In *COLING 2000: the 18th Conference on Computational Linguistics*, vol. 1. Saarbrücken. DOI: 10 . 3115 / 990820.990830.

Bod, Rens. 2007. Unsupervised syntax-based machine translation: The contribution of discontiguous phrases. In *Proceedings of the Machine Translation Summit XI*, 51–56. Copenhagen.

Bod, Rens & Ronald M. Kaplan. 1998. A probabilistic corpus-driven model for lexical-functional analysis. In *ACL '98/COLING '98: Proceedings of the 36th annual meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, vol. 2, 145–151. Montréal: Association for Computational Linguistics. DOI: 10.3115/980451.980869.

Bod, Rens & Ronald M. Kaplan. 2003. A Data-Oriented Parsing model for Lexical-Functional Grammar. In Rens Bod, Remko Scha & Ronald M. Kaplan (eds.), *Data-oriented parsing*. Stanford, CA: CSLI Publications.

Brants, Sabine, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius & George Smith. 2002. The TIGER treebank. In *Proceedings of the 1st Workshop on Treebanks and Linguistic Theories*, 24–41.

Burke, Michael. 2006. *Automatic annotation of the Penn-II Treebank with f-structure information*. Dublin: School of Computing, Dublin City University. (Doctoral dissertation).

Burke, Michael, Aoife Cahill, Mairéad McCarthy, Ruth O'Donovan, Josef van Genabith & Andy Way. 2004. Evaluating automatic LFG f-structure annotation for the Penn-II treebank. *Research on Language and Computation* 2(4). 523–547. DOI: 10.1007/s11168-004-7428-y.

Burke, Michael, Olivia S.-C. Lam, Aoife Cahill, Rowena Chan, Ruth O'Donovan, Adams Bodomo, Josef van Genabith & Andy Way. 2004. Treebank-based acquisition of a Chinese lexical-functional grammar. In *Proceedings of the 18th Pacific Asia conference on language, information and computation*, 161–172. Tokyo.

Butt, Miriam. 1994. Machine translation and complex predicates. In *Konferenz zur Verarbeitung natürlicher Sprache (KONVENS 94)*, 62–71. Vienna.

Cahill, Aoife. 2004. *Parsing with automatically acquired, wide-coverage, robust, probabilistic LFG approximations*. Dublin: School of Computing, Dublin City University. (Doctoral dissertation).

Cahill, Aoife, Michael Burke, Ruth O'Donovan, Stefan Riezler, Josef van Genabith & Andy Way. 2008. Wide-coverage deep statistical parsing using automatic dependency structure annotation. *Computational Linguistics* 34(1). 81–124. DOI: 10.1162/coli.2008.34.1.81.

Cahill, Aoife, Michael Burke, Ruth O'Donovan, Josef van Genabith & Andy Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage PCFG-based LFG approximations. In *Proceedings of the 42nd annual meeting of the Association for Computational Linguistics*, 319–326. Barcelona: Association for Computational Linguistics. DOI: 10.3115/1218955.1218996.

Cahill, Aoife, Martin Forst, Michael Burke, Mairéad McCarthy, Ruth O'Donovan, Christian Rohrer, Josef van Genabith & Andy Way. 2005. Treebank-based acquisition of multilingual unification grammar resources. *Journal of Research on Language and Computation; Special Issue on "Shared Representations in Multilingual Grammar Engineering"* 3(2–3). 247–279. DOI: 10.1007/s11168-005-1296-y.

Cahill, Aoife, Martin Forst, Mairéad McCarthy, Ruth O'Donovan, Christian Rohrer, Josef van Genabith & Andy Way. 2003. Treebank-based multilingual unification-grammar development. In *Proceedings of the workshop on ideas and strategies for multilingual grammar development at the 15th European Summer School in Logic, Language and Information*, 17–24. Vienna.

Cahill, Aoife, John T. Maxwell III, Paul Meurer, Christian Rohrer & Victoria Rosén. 2008. Speeding up LFG parsing using c-structure pruning. In *COLING 2008: Proceedings of the workshop on Grammar Engineering Across Frameworks*, 33–40. Manchester. DOI: 10.3115/1611546.1611551.

Cahill, Aoife, Mairéad McCarthy, Josef van Genabith & Andy Way. 2002a. Automatic annotation of the Penn-Treebank with LFG F-structure information. In *LREC'02 workshop on linguistic knowledge acquisition and representation: Bootstrapping annotated language data*, 76–95. Las Palmas.

Cahill, Aoife, Mairéad McCarthy, Josef van Genabith & Andy Way. 2002b. Parsing with PCFGs and automatic f-structure annotation. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '02 conference*, 76–95. Stanford, CA: CSLI Publications.

Carroll, John, Edward Briscoe & Antonio Sanfilippo. 1998. Parser evaluation: A survey and new proposal. In *Proceedings of the international conference on language resources and evaluation*, 447–454. Granada.

Carroll, John & Ted Briscoe. 2004. High precision extraction of grammatical relations. In Harry Bunt, John Carroll & Giorgio Satta (eds.), *New developments in parsing technology* (Text, Speech and Language Technology 23). Dordrecht: Springer. DOI: 10.1007/1-4020-2295-6_3.

Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* 2(3). 113–124. DOI: 10.1109/tit.1956.1056813.

Chomsky, Noam. 1981. *Lectures on government and binding*. Dordrecht: Foris Publications. DOI: 10.1515/9783110884166.

Chrupała, Grzegorz & Josef van Genabith. 2006. Improving treebank-based automatic LFG induction for Spanish. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '06 conference*. Stanford, CA: CSLI Publications.

Civit, Montserrat. 2003. *Criterios de etiquación y desambiguación morfosintáctica de corpus en español*. Barcelona: Universitat Politècnica de Catalunya. (Doctoral dissertation).

Collins, Michael. 1999. *Head-Driven statistical models for natural language parsing*. Philadelphia: University of Pennsylvania. (Doctoral dissertation).

Cormons, Boris. 1999. *Analyse et désambiguisation: Une approche à base de corpus (Data-Oriented Parsing) pour les répresentations lexicales fonctionnelles*. Rennes: Université de Rennes. (Doctoral dissertation).

Dalrymple, Mary, Ronald M. Kaplan, John T. Maxwell III & Annie Zaenen (eds.). 1995. *Formal issues in Lexical-Functional Grammar*. Stanford, CA: CSLI Publications.

Frank, Anette. 2000. Automatic f-structure annotation of treebank trees. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '00 conference*, 140–160. Stanford, CA: CSLI Publications.

Futrell, Richard, Laura Stearns, Daniel L. Everett, Steven T. Piantadosi & Edward Gibson. 2016. A corpus investigation of syntactic embedding in Pirahã. *PLoS ONE* 11(3). DOI: 10.1371/journal.pone.0145289.

Gazdar, Gerald & Chris Mellish. 1989. *Natural language processing in PROLOG: An introduction to computational linguistics*. Wokingham, UK: Addison-Wesley.

Graham, Yvette & Josef van Genabith. 2012. Exploring the parameter space in statistical machine translation via f-structure transfer. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '12 conference*, 254–270. Stanford, CA: CSLI Publications.

Guo, Yuqing. 2009. *Treebank-based acquisition of Chinese LFG resources for parsing and generation*. Dublin: School of Computing, Dublin City University. (Doctoral dissertation).

Guo, Yuqing, Josef van Genabith & Haifeng Wang. 2007. Treebank-based acquisition of LFG resources for Chinese. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '07 conference*, 214–232. Stanford, CA: CSLI Publications.

Hearne, Mary. 2005. *Data-oriented models of parsing and translation*. Dublin: School of Computing, Dublin City University. (Doctoral dissertation).

Hearne, Mary & Khalil Sima'an. 2004. Structured parameter estimation for LFG-DOP. In Nicolas Nicolov, Kalina Bontcheva, Galia Angelova & Ruslan Mitkov (eds.), *Recent advances in natural language processing III: Selected papers from RANLP 2003* (Current Issues in Linguistic Theory 260), 183–192. Amsterdam: John Benjamins. DOI: 10.1075/cilt.260.20hea.

Hockenmaier, Julia & Mark Steedman. 2002. Generative models for statistical parsing with combinatory categorial grammar. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 335–342. Philadelphia.

Kaplan, Ronald M. & Joan Bresnan. 1982. Lexical-Functional Grammar: A formal system for grammatical representation. In Joan Bresnan (ed.), *The mental representation of grammatical relations*, 173–281. Cambridge, MA: The MIT Press. Reprinted in Dalrymple, Kaplan, Maxwell & Zaenen (1995: 29–130).

Kaplan, Ronald M., John T. Maxwell III, Tracy Holloway King & Richard Crouch. 2004. Integrating finite-state technology with deep LFG grammars. In *Proceedings of the workshop on combining shallow and deep processing for NLP at the European summer school on logic, language, and information (ESSLLI)*.

Kaplan, Ronald M., Klaus Netter, Jürgen Wedekind & Annie Zaenen. 1989. Translation by structural correspondences. In *Proceedings of the 4th conference of the European chapter of the ACL (EACL 1989)*, 272–281. Association for Computational Linguistics. DOI: 10.3115/976815.976852. Reprinted in Dalrymple, Kaplan, Maxwell & Zaenen (1995: 311–330).

Kaplan, Ronald M. & Jürgen Wedekind. 1993. Restriction and correspondence-based translation. In *Proceedings of the 6th conference of the European chapter of the ACL (EACL 1993)*, 193–202. Association for Computational Linguistics. DOI: 10.3115/976744.976768.

Kay, Martin, Jean Mark Gawron & Peter Norvig. 1994. *Verbmobil: A translation system for face-to-face dialog*. Stanford, CA: CSLI Publications.

King, Tracy Holloway, Richard Crouch, Stefan Riezler, Mary Dalrymple & Ronald M. Kaplan. 2003. The PARC 700 Dependency Bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora, held at the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*. Budapest: Association for Computational Linguistics.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the Machine Translation Summit X*, 79–86. Phuket, Thailand.

Koehn, Philipp, Franz Josef Och & Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 127–133. Edmonton. DOI: 10.21236/ada461156.

Kurohashi, Sadao & Makoto Nagao. 1997. Kyoto Daigaku text corpus project. In *Proceedings of the third annual meeting of the association of applied natural language processing*, 115–118. In Japanese.

Lappin, Shalom, Igal Golan & Mori Rimon. 1989. *Computing grammatical functions from configurational parse trees*. Technical Report 88.268. Haifa: IBM Israel.

Maamouri, Mohamed & Ann Bies. 2004. Developing an Arabic treebank: Methods, guidelines, procedures, and tools. In *Proceedings of the workshop on computational approaches to Arabic script-based languages*, 2–9. Geneva. DOI: 10.3115/1621804.1621808.

Marcus, Mitchell, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz & Britta Schasberger. 1994. The Penn treebank: Annotating predicate argument structure. In *Proceedings of the workshop on human language technology*, 114–119. Plainsboro, NJ. DOI: 10.3115/1075812.1075835.

McCarthy, Mairéad. 2003. *Design and evaluation of the linguistic basis of an automatic f-structure annotation algorithm for the Penn-II Treebank*. Dublin: School of Computing, Dublin City University. (MA thesis).

Miyao, Yusuke, Takashi Ninomiya & Jun'ichi Tsujii. 2003. Probabilistic modeling of argument structures including non-local dependencies. In *Proceedings of the conference on recent advances in natural language processing (RANLP 2003)*, 285–291. Borovets, Bulgaria.

O'Donovan, Ruth. 2006. *Automatic extraction of large-scale multilingual lexical resources*. Dublin: School of Computing, Dublin City University. (Doctoral dissertation).

O'Donovan, Ruth, Michael Burke, Aoife Cahill, Josef van Genabith & Andy Way. 2004. Large-scale induction and evaluation of lexical resources from the Penn-II Treebank. In *Proceedings of the 42nd annual meeting of the Association for Computational Linguistics*, 368–375. Barcelona. DOI: 10.3115/1218955.1219002.

O'Donovan, Ruth, Aoife Cahill, Josef van Genabith & Andy Way. 2005. Automatic acquisition of Spanish LFG resources from the CAST3LB treebank. In Miriam

Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '05 conference*, 334–352. Stanford, CA: CSLI Publications.

Oya, Masanori & Josef van Genabith. 2007. Automatic acquisition of Lexical-Functional Grammar resources from a Japanese dependency corpus. In *Proceedings of the 21st Pacific Asia conference on language, information and computation*, 375–384. Seoul.

Poutsma, Arjen. 2000. Data-oriented translation. In *COLING 2000: the 18th Conference on Computational Linguistics*, vol. 2, 635–641. Saarbrücken. DOI: 10.3115/992730.992738.

Rehbein, Ines & Josef van Genabith. 2009. Automatic acquisition of LFG resources for German – As good as it gets. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '09 conference*, 480–500. Stanford, CA: CSLI Publications.

Riezler, Stefan, Tracy Holloway King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell III & Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and discriminative estimation techniques. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. Philadelphia: Association for Computational Linguistics.

Riezler, Stefan & John T. Maxwell III. 2006. Grammatical machine translation. In *Proceedings of the human language technology conference of the NAACL, main conference*, 248–255. New York: Association for Computational Linguistics. DOI: 10.3115/1220835.1220867.

Rimell, Laura, Stephen Clark & Mark Steedman. 2009. Unbounded dependency recovery for parser evaluation. In *Proceedings of the 2009 conference on Empirical Methods in Natural Language Processing*, 813–821. Singapore. DOI: 10.3115/1699571.1699619.

Sadler, Louisa & Henry Thompson. 1991. Structural non-correspondence in translation. In *Proceedings of the 5th conference of the European chapter of the ACL (EACL 1991)*, 293–298. Berlin: Association for Computational Linguistics. DOI: 10.3115/977180.977231.

Sadler, Louisa, Josef van Genabith & Andy Way. 2000. Automatic f-structure annotation from the AP treebank. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '00 conference*, 226–243. Stanford, CA: CSLI Publications.

Schluter, Natalie. 2011. *Treebank-based deep grammar acquisition for French probabilistic parsing resources*. Dublin: School of Computing, Dublin City University. (Doctoral dissertation).

Schluter, Natalie & Josef van Genabith. 2007. Preparing, restructuring, and augmenting a French treebank: Lexicalised parsers or coherent treebanks? In *Proceedings of the 10th Pacific Asia Conference on Language, Information and Computation*, 200–209. Melbourne.

Schluter, Natalie & Josef van Genabith. 2008. Treebank-based acquisition of LFG parsing resources for French. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC'08)*, 2909–2916. Marrakech.

Sima'an, Khalil. 1997. An optimized algorithm for Data-Oriented Parsing. In Ruslan Mitkov & Nicolas Nicolov (eds.), *Recent advances in natural language processing: Selected papers from RANLP '95* (Current Issues in Linguistic Theory), 35–46. Amsterdam: John Benjamins. DOI: 10.1075/cilt.136.05sim.

Telljohann, Heike, Erhard W. Hinrichs, Sandra Kübler, Heike Zinsmeister & Kathrin Beck. 2006. *Stylebook for the Tübingen treebank of written German (TüBa-D/Z)*. Tübingen: Universität Tübingen.

Tounsi, Lamia, Mohammed Attia & Josef van Genabith. 2009a. Automatic treebank-based acquisition of Arabic LFG dependency structures. In *Proceedings of the EACL 2009 workshop on computational approaches to Semitic languages*, 45–52. Athens: Association for Computational Linguistics. DOI: 10.3115/1621774.1621783.

Tounsi, Lamia, Mohammed Attia & Josef van Genabith. 2009b. Parsing Arabic using treebank-based LFG resources. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '09 conference*, 583–586. Stanford, CA: CSLI Publications.

van Genabith, Josef, Anette Frank & Michael Dorna. 1998. Transfer constructors. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '98 conference*, 190–205. Stanford, CA: CSLI Publications.

van Genabith, Josef, Louisa Sadler & Andy Way. 1999. Deriving an LFG from a treebank resource. In *Proceedings of the ATALA international workshop on treebanks*, 107–114. Paris.

van Genabith, Josef, Andy Way & Louisa Sadler. 1999a. Data-driven compilation of LFG semantic forms. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora (LINC-99)*, 69–76. Bergen: Association for Computational Linguistics.

van Genabith, Josef, Andy Way & Louisa Sadler. 1999b. Semi-automatic generation of f-structures from treebanks. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '99 conference*, 19–21. Stanford, CA: CSLI Publications.

Way, Andy. 1999. A hybrid architecture for robust MT using LFG-DOP. *Journal of Experimental and Theoretical Artificial Intelligence, Special Issue on Memory-Based Learning* 11. 441–471. DOI: 10.1080/095281399146490.

Way, Andy. 2001. *A hybrid architecture for robust MT*. Colchester, UK: University of Essex. (Doctoral dissertation).

Way, Andy. 2003. Machine translation using LFG-DOP. In Rens Bod, Remko Scha & Khalil Sima'an (eds.), *Data-oriented parsing*, 359–384. Stanford, CA: CSLI Publications.

Way, Andy, Ian Crookston & Jane Shelton. 1997. A typology of translation problems for Eurotra translation machines. *Machine Translation* 12. 323–374.

Xia, Fei. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, 398–403. Peking.

Xue, Nianwen, Fu-Dong Chiou & Martha Palmer. 2002. Building a large-scale annotated Chinese corpus. In *COLING 2002: the 19th International Conference on Computational Linguistics*. Taipei. DOI: 10.3115/1072228.1072373.

Zaenen, Annie & Ronald M. Kaplan. 1995. Formal devices for linguistic generalizations: West Germanic word order in LFG. In Jennifer S. Cole, Georgia M. Green & Jerry L. Morgan (eds.), *Linguistics and computation*, 3–27. Stanford, CA: CSLI Publications. Reprinted in Dalrymple, Kaplan, Maxwell & Zaenen (1995: 215–240).