# Structured Parameter Estimation for LFG-DOP using Backoff

**Mary Hearne**$^\star$ and **Khalil Sima'an**$^\dagger$

$\star$ School of Computing, Dublin City University, Glasnevin, Dublin 9, Ireland

† Institute for Logic, Language and Computation, University of Amsterdam
Nieuwe Achtergracht 166, 1018 WV Amsterdam, The Netherlands
simaan@science.uva.nl; mhearne@computing.dcu.ie

## Abstract

Despite its state-of-the-art performance, the Data Oriented Parsing (DOP) model has been shown to suffer from biased parameter estimation, and the good performance seems more the result of ad hoc adjustments than correct probabilistic generalization over the data. In recent work, we developed a new estimation procedure, called Backoff Estimation, for DOP models that are based on Phrase-Structure annotations (so called Tree-DOP models). Backoff Estimation deviates from earlier methods in that it treats the model parameters as a highly structured space of correlated events (backoffs), rather than a set of disjoint events. In this paper we show that the problem of biased estimates also holds for DOP models that are based on Lexical-Functional Grammar annotations (i.e. LFG-DOP), and that the LFG-DOP parameters also constitute a hierarchically structured space. Subsequently, we adapt the Backoff Estimation algorithm from Tree-DOP to LFG-DOP models. Backoff Estimation turns out to be a natural solution to some of the specific problems of robust parsing under LFG-DOP.

The DOP model (Bod, 1995; Bod, 2001) currently exhibits good performance on current benchmark treebanks, e.g. (Marcus et al., 1993). These treebanks are annotated with impoverished phrase-structure parse-trees and the DOP model that fits these annotations, called Tree-DOP, works with the common rewrite operation of substitution ( inherited from Context-Free Grammars). Despite the simplicity of the rewrite formalism underlying Tree-DOP, probability estimation turns out not to be as straightforward as it initially seemed. Earlier studies (Johnson, 2002; Sima'an and Buratto, 2003) have shown the bias of the three previous estimation procedures for Tree-DOP, namely (Bod, 1995), (Bonnema et al., 1999) and (Bod, 2001). Recently, a new study (Sima'an and Buratto, 2003) shows that the main problem with estimation for Tree-DOP is that the various parameters cannot be assumed to be disjoint, as earlier work does; in fact, the Tree-DOP parameter space is shown to abide by a partial order that structures these parameters according to correlations of occurrence between the different parameters. A suitable algorithm, Backoff Estimation, takes into account this fact during the parameter estimation for Tree-DOP. Preliminary experiments show improved performance, despite the impoverished first implementation.

In this paper, we study parameter estimation for the extension of DOP to linguistic annotations that are richer than phrase-structure. We concentrate on the extension of DOP to Lexical-Functional Grammar (LFG) annotations, i.e. LFG-DOP (Bod and Kaplan, 2003). Naturally, the problem of biased parameter estimation carries over from Tree-DOP to LFG-DOP. In fact, the bias in Tree-DOP is further compounded with specific aspects of LFG-DOP that allow for robust processing that can be achieved by abstraction over actually occurring treebank structures. We show how the Backoff Estimation procedure applies to LFG-DOP and discuss the resulting model. It turns out that Backoff Estimation naturally realizes the specific model architecture that has been observed to work best in previous experiments with LFG-DOP (Bod and Kaplan, 1998; Bod and Kaplan, 2003).

Section 1 provides a review of the Tree-DOP model. Section 2 reviews the Backoff Estimation algorithm. Section 3 reviews LFG-DOP and section 4 extends the Backoff Estimation algorithm to LFG-DOP. Finally, section 5 provides the conclusions from this work.

## 1 Tree-DOP: Phrase-Structure

Like other treebank models, Tree-DOP extracts a finite set of rewrite productions, called *subtrees*, from the training treebank together with probabilities. A connected subgraph of a treebank tree $t$ is called a *subtree* iff it consists of one or more context-free productions[1] from $t$. Following (Bod, 1995), the set of rewrite productions of Tree-DOP consists of *all* the subtrees of the treebank trees. Figure 3 exemplifies the set of subtrees extracted from the treebank of Figure 1.

[1] Note that a non-leaf node labeled $p$ in tree $t$ dominating a sequence of nodes labeled $c_1, \cdots, c_n$ consists of a graph that represents the context-free production: $p \rightarrow c_1 \cdots c_n$.

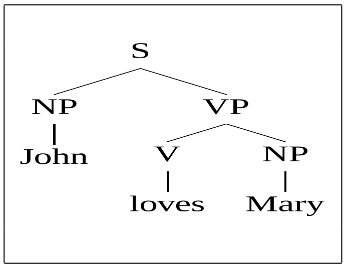

Figure 1: A toy treebank

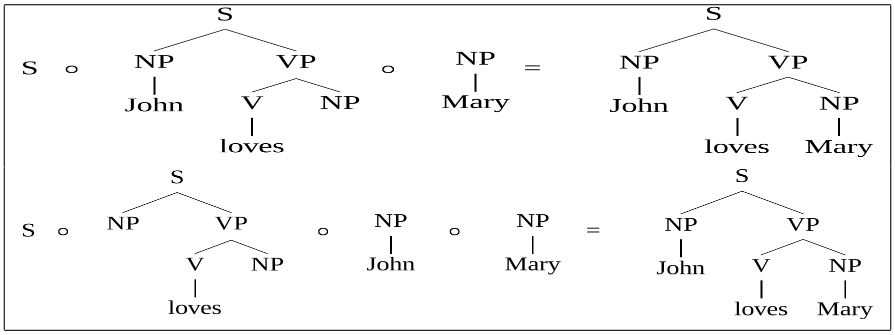

Figure 2: Two different derivations of the same parse

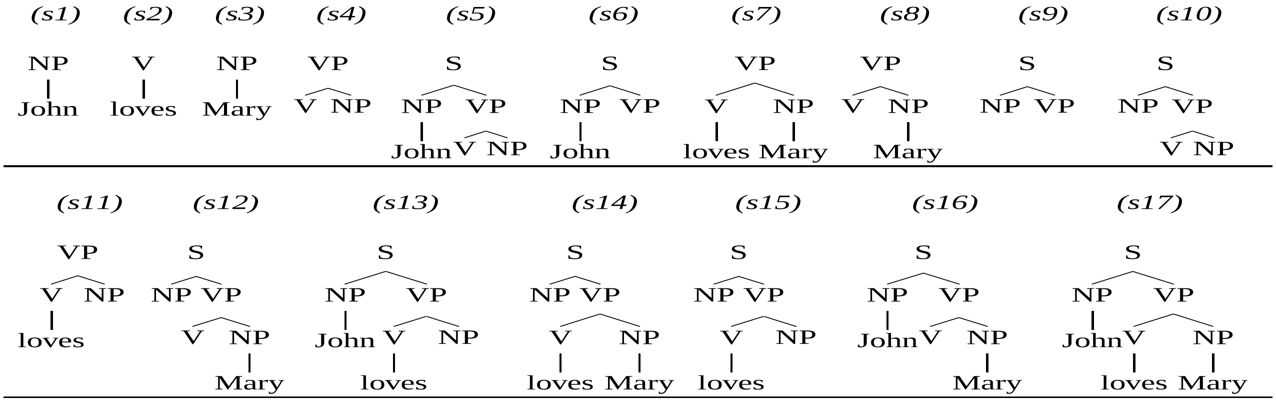

Figure 3: The subtrees of the treebank in Figure 1

The Tree-DOP model employs the set of subtrees as a Stochastic Tree-Substitution Grammar (STSG): a TSG is a rewrite system similar to Context-Free Grammars (CFGs) ; the only difference is that the productions of a TSG are subtrees of arbitrary depth[2]. A TSG derivation proceeds by combining subtrees using the substitution operation ∘ starting from the start symbol $S$ of the TSG. In contrast with CFG derivations, multiple TSG derivations may generate the same parse. For example, the parse in Figure 1 can be derived at least in two different ways as shown in Figure 2. In this sense, the Tree-DOP model deviates from other contemporary models, e.g. (Chelba and Jelinek, 1998; Charniak, 2000), that belong to the so-called History-Based Stochastic Grammar (HBSG) family (Black et al., 1993). The latter models generate every parse-tree through a unique stochastic derivation.

A Stochastic TSG (STSG) is a TSG extended with a probability mass function $P$ over the set of subtrees: the probability of subtree $t$, that has root label $\mathtt{R}_t$, is given by $P(t|\mathtt{R}_t)$, i.e. for every non-terminal $A$: $\sum_{\{t|\mathtt{R}_t=A\}} P(t \mid A) = 1$.

Given a probability function $P$, the probability of a derivation $D = S \circ t_1 \circ \cdots \circ t_n$ is defined by $P(D \mid S) = \prod_{i=1}^{n} P(t_i|\mathtt{R}_{t_i})$. The probability of a parse is defined by the sum of the probabilities of all derivations in the STSG that generate that parse.

When parsing an input sentence $U$ under a Tree-DOP model, the preferred parse $T$ is the Most Probable Parse (MPP) for that sentence: $\arg\max_T P(T|U)$. However, the problem of computing the MPP is known to be intractable (Sima'an, 2002). In contrast, the calculation of the Most Probable Derivation (MPD) $D$ for the input sentence $U$ i.e., $\arg\max_D P(D|U)$, can be done in time cubic in sentence length.

The problem of how to estimate the probabilities of the subtrees from a treebank is *not* as straightforward as originally thought. So far, there exist three estimation procedures (Bod, 1995; Bonnema et al., 1999; Bod, 2001). As shown in (Bonnema et al., 1999; Johnson, 2002; Sima'an and Buratto, 2003), all three estimation procedures turn out to be biased in an unintuitive manner.

[2] The depth of a tree is the number of edges along the longest path from the root to a leaf node.

## 2 Backoff Estimation for DOP

For the sake of completeness we review in this section the Backoff Estimation procedure presented in (Sima'an and Buratto, 2003).

Consider the common situation where a subtree[3] $t$ is equal to a tree generated by a derivation $t_1 \circ \cdots \circ t_n$ involving multiple subtrees $t_1 \cdots t_n$. For example, subtree *s17* (Figure 3) can be constructed by different derivations such as *(s16 ∘ s2)*, *(s14 ∘ s1)* and *(s15 ∘ s1 ∘ s3)*. We will refer to subtrees that can be constructed from derivations involving other subtrees with the term *complex subtrees*.

We observe that because the statistics for these different derivations come from the treebank, these statistics must be correlated. Next we will characterize these correlations in order to arrive at a suitable estimation procedure.

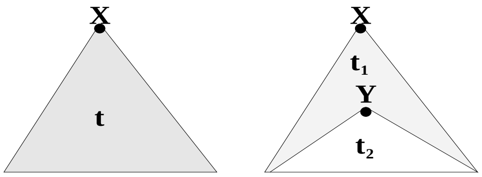

Figure 4: Sketch of two derivation of the same subtree

For every complex subtree $t$, we restrict our attention only to the derivations involving pairs of subtrees; in other words, we focus on subtree $t$ such that there exist subtrees $t_1$ and $t_2$ such that $t = (t_1 \circ t_2)$ (see figure 4). In DOP, the probability of $t$ is given by $P(t|\mathtt{R}_t)$. In contrast, the derivation probability is given by $P(t_1|\mathtt{R}_{t_1})P(t_2|\mathtt{R}_{t_2})$. However, according to the chain rule $P(t_1 \circ t_2|\mathtt{R}_{t_1}) = P(t_1|\mathtt{R}_{t_1})P(t_2|t_1)$. Therefore, the derivation $t_1 \circ t_2$ embodies an independence assumption realized by the approximation[4]: $P(t_2|t_1) \approx P(t_2|\mathtt{R}_{t_2})$. This approximation involves a so-called *backoff*, i.e. a weakening of the conditioning context from $P(t_2|t_1)$ to $P(t_2|\mathtt{R}_{t_2})$. Hence, we will say that the derivation $t_1 \circ t_2$ constitutes *a backoff* of subtree $t$ and we will write ($t \geq_{bfk} t_1 \circ t_2$) to express this fact.

The backoff relation $\geq_{bfk}$ between a subtree and a pair of other subtrees allows for a partial order between the derivations of the subtrees extracted from a treebank. A graphical representation of this partial order is a directed acyclic graph which consists of a node for each pair of subtrees $t_i, t_j$ that constitute a derivation of another complex subtree. A directed edge points from a subtree $t_i$ in a node[5] to another node containing a pair of subtrees $\langle t_j, t_k \rangle$ iff $t_i \geq_{bfk} t_j \circ t_k$. We refer to this graph as the *backoff graph*. An example based on the subtrees of Figure 3 is shown in Figure 5, where $s0$ stands for a subtree consisting of a single node labeled $S$ (the start symbol). We distinguish two sets of subtrees: initial and atomic. Initial subtrees are subtrees that do not participate in a backoff derivation of any other subtree. In Figure 3, subtree $s17$ is the only initial subtree.Atomic subtrees are subtrees for which there are no backoffs. In Figure 3, these are subtrees of depth one (double circled in the backoff graph).

In the DOP model (under any known estimation procedure), the probablity of a parse-tree is defined as the sum of the probabilities of all derivations that generate this parse-tree. This means that DOP linearly interpolates derivations involving subtrees from different levels of the backoff graph; this is similar to the way Hidden Markov Models interpolate different Markov orders over, e.g. words, for calculating sentence probability. Hence, we will refer to the different levels of subtrees in the backoff graph as the *Markov orders*.

**Backoff DOP** Crucially, the partial order over the subtrees, embodied in the backoff graph, can be exploited for turning DOP into a "backedoff model" as follows. A subtree is generated by a sequence of derivations *ordered by the backoff relation*. This is in sharp contrast with existing DOP models that consider the different derivations leading to the same subtree as a set of *disjoint* events. Next we present the estimation procedure that accompanies this new realization of DOP as a recursive backoff over the different Markov orders.

**Estimation vs. smoothing** It is common in probabilistic modeling to *smooth* a probability distribution $P(t|X, Y)$ by a backoff distribution thereof e.g. $P(t|X)$. The smoothing of $P(t|X, Y)$ aims at dealing with the problem of sparse data (whenever the probability $P(t|X, Y)$ is zero). The backoff value $P(t|X)$ can be used as an approximation of $P(t|X, Y)$ under the assumption that $t$ and $Y$ are independent. Smoothing, then, aims at enlarging the space of non-zero events in the distribution $P(t|X, Y)$. Hence, the goal of smoothing differs from our goal. While smoothing aims at filling the zero gaps in a distribution, our goal is to estimate the distribution (a priori to smoothing it). Despite these differences, we employ a back-

[3] The term "subtree" is reserved for the tree-structures that DOP extracts from the treebank.

[4] Note that $\mathtt{R}_{t_2}$ is part of $t_1$ (the label of the substitution site).

[5] In a pair $\langle t_h, t_i \rangle$ or $\langle t_i, t_h \rangle$ that constitutes a node.

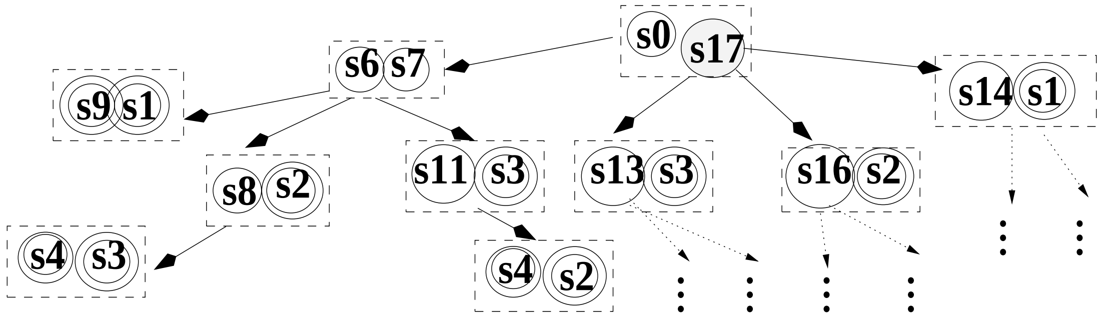

Figure 5: A portion of the backoff graph for the subtrees in Figure 3

off method for parameter estimation by *redistributing probability mass among DOP model subtrees*.

**Katz Backoff** The Katz Backoff method (Katz, 1987; Chen and Goodman, 1998) is a smoothing technique based on the discounting method of Good-Turing (GT) (Good, 1953; Chen and Goodman, 1998). Given a higher order distribution $P(t|X,Y)$, Katz backoff employs the GT formula for discounting from this distribution leading to $P_{GT}(t|X,Y)$. Then, the probability mass that was discounted $(1 - \sum_t P_{GT}(t|X,Y))$ is distributed over the lower order distribution $P(t|X)$.

**Estimation by Backoff** We assume initial probability estimates $P_f$ based on frequency counts as in $DOP_{rf}$ (Bod, 1995). The present backoff estimation procedure operates iteratively, top-down over the backoff graph, starting with the initial and moving down to the atomic subtrees. In essence this procedure *transfers, stepwisely, probability mass* from complex subtrees to their backoffs.

Let $P^c$ represent the current probability estimate resulting from $i$ previous steps of re-estimation (initially, at step $i = 0$, $P^0 := P_f$). After $i$ steps, the edges of the backoff graph lead to the *current layer* of nodes. For every $t$, a subtree in a node from the current layer in the backoff graph, an edge $e$ outgoing from $t$ stands for the relation $(t \geq_{bfk} t_1 \circ t_2)$, where $\langle t_1, t_2 \rangle$ is the node at the other end of edge $e$. We know that

$$P^c(t|\mathtt{R}_t) = P^c(t_1|\mathtt{R}_{t_1})P^c(t_2|t_1)$$

$$P^c(t_1 \circ t_2) = P^c(t_1|\mathtt{R}_{t_1})P^c(t_2|\mathtt{R}_{t_2})$$

This mean that $P^c(t_2|t_1)$ is backedoff to $P^c(t_2|\mathtt{R}_{t_2})$. Hence, we may adapt the Katz method to estimate the Backoff DOP probability $P_{bo}$ as follows:

$$P_{bo}(t_2|t_1) \quad =$$

$$\begin{cases} P^c_{GT}(t_2|t_1) \;+\; \alpha(t_1)\; P_f(t_2|\mathtt{R}_{t_2}) & [\mathtt{P^c(t_2|t_1)} > 0] \\ \alpha(t_1)\; P_f(t_2|\mathtt{R}_{t_2}) & \mathtt{otherwise} \end{cases}$$

where $\alpha(t_1)$ is a normalization factor that guarantees that the sum of the probabilities of subtrees with the same root label is one. Simple arithmetic leads to the following formula:

$$\alpha(t_1) = 1 - \sum_{t_2 : f(t_1, t_2) > 0} P^c_{GT}(t_2|t_1)$$

Using the above estimate of $P_{bo}(t_2|t_1)$, the other backoff estimates are calculated as follows:

$$P_{bo}(t|\mathtt{R}_t) := P_f(t_1|\mathtt{R}_{t_1})\; P^c_{GT}(t_2|t_1)$$

$$P_{bo}(t_1|\mathtt{R}_{t_1}) := (\; 1 + \alpha(t_1)\;)\; P_f(t_1|\mathtt{R}_{t_1})$$

Before the next step $i + 1$ of Katz backoff takes place over the next layer in the backoff graph, the current probabilities are updated as follows: $P^{i+1}(t_1|\mathtt{R}_{t_1}) := P_{bo}(t_1|\mathtt{R}_{t_1})$.

Note that $P_{bo}$ is a proper distribution in the sense that for all nonterminals $A$: $\sum_t P(t|A) = 1$. This is guaranteed by the redistribution of the reserved probability mass at every step of the procedure over the layers of the backoff graph. Furthermore, we note that the present method is *not* a smoothing method since it applies Katz Backoff for redistributing probability mass *only* among subtrees that *did occur* in the tree-bank. The present method does not address probability estimation for unknown/unseen events.

**Summary of experiments:** In (Sima'an and Buratto, 2003) we describe a first implementation of Backoff estimation for DOP and report on a series of experiments. Because of the large number of DOP subtrees the current implementation applies Backoff estimation only to $t \geq_{bfk} t_1 \circ t_2$ iff $t_2$ is a lexical subtree i.e., $t_2 = X \rightarrow w$ where $X$ is a Part of Speech (PoS) tag and $w$ a word. This recognizes the importance of good estimates of the probabilities of lexicalized subtrees. The experiments on the Dutch OVIS tree-bank show that the Backoff estimation method

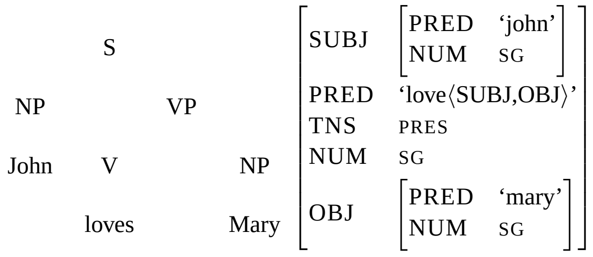

Figure 6: An LFG representation for *John loves Mary*.

described above outperforms the original estimator (subtree relative frequency) used in (Bod, 1995; Bod, 2001). The improved behavior has been validated on multiple different train/test splits. Furthermore, both the new Backoff estimator and the subtree relative frequency estimator outperform the estimator suggested in (Bonnema et al., 1999).

Next we extend the new Backoff estimator to LFG-DOP, and show that this estimator solves some of the hard problems that LFG-DOP suffers from.

## 3 LFG-DOP: Lexical-Functional Grammar

Before we show how Backoff Estimation can be applied to estimate the model parameters, we specify the four elements of LFG-DOP:

**Representation:** Lexical Functional Grammar (LFG) (Bresnan, 2001) is a constraint-based theory of language which aims to analyse language in lexical and functional terms rather than solely in terms of phrase structure. The LFG-DOP model (Bod and Kaplan, 1998) employs representations, such as the one in Figure 6, which comprise two parallel levels, constituent structure (c-structure) and functional structure (f-structure), and a mapping between them. The c-structures describe surface structure, the f-structures describe grammatical relations and $\phi$ maps between these levels of linguistic representation. This relationship is generally expressed as a triple of the form $<$c,$\phi$,f$>$. An f-structure unit $f$ is $\phi$-accessible from a node $n$ if either $f$ is linked to $n$ or $f$ is contained within an f-structure linked to $n$. This reflects the intuitive idea that nodes can only access information in the units of f-structure to which they are $\phi$-linked.

**Decomposition into fragments:** LFG-DOP fragments consist of connected subtrees (as in Tree-DOP) whose nodes are in $\phi$ correspondence with (partial) f-structures. In operational terminology, the Tree-DOP decomposition of a treebank tree into subtrees proceeds using two operators: *root* and *frontier*. These operators select the nodes that delimit a subtree: *root* selects the root node, and *frontier* selects the frontier nodes. The connected subgraph between these selected nodes constitutes the subtree.

For LFG-DOP, these operators are extended as follows (Bod and Kaplan, 2003). When a node is selected by the *root operation*, all nodes outside that node's subtree are erased, as in Tree-DOP. Further, all $\phi$-links coming from the erased nodes are removed and all f-structure units not $\phi$-accessible from the remaining nodes are erased. The *root* operation also deletes the PRED attributes (semantic forms) local to f-structures corresponding to erased nodes. The *frontier* operation selects a set of frontier nodes and deletes all subtrees they dominate, also removing the $\phi$-links and semantic forms (but not features) corresponding to any deleted nodes. An example is given in Figure 7.

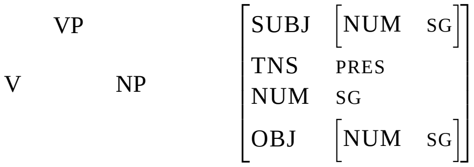

Figure 7: A fragment generated by *root* and *frontier* from the representation in Figure 6.

**Composition:** LFG-DOP derivations proceed using composition operations that extend the substitution operation of Tree-DOP. The LFG-DOP composition operation (∘) involves two stages: c-structures are combined exactly as in Tree-DOP and their corresponding f-structures are unified recursively.

According to LFG theory (Bresnan, 2001), c-structures and f-structures must satisfy certain well-formedness conditions: *uniqueness* specifies that each attribute in the f-structure can have at most one value, *coherence* prohibits the presence of grammatical functions which are not required by the lexical predicate, and *completeness* requires that all functions governed by a lexical predicate must be present in its f-structure. Any

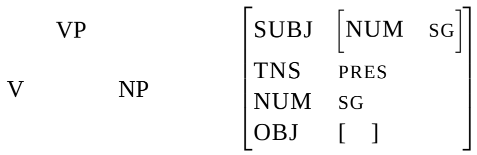

Figure 8: A fragment generated by the *discard* operation from the representation in Figure 7.

parse generated by a sequence of composition operations must satisfy these conditions.

**Probability Model:** As with Tree-DOP, the probability of a derivation is the joint probability of choosing each fragment involved in that derivation. Hence, the probability of a derivation is the product of the probabilities of choosing each of the fragments involved in that derivation $P(f_1 \circ ... \circ f_n) = \prod_i P(f_i)$ and the probability of a parse T is the sum of the probabilities of its distinct derivations $P(T) = \sum_{D\ derives\ T} P(D)$.

In Tree-DOP, each Competition Set (CS) simply contains all fragments with the same root node because there are no conditions to be met other than those enforced at each derivation step by the composition operator. However, the definition of the competition set for LFG-DOP depends on which of the LFG well-formedness conditions are enforced on-line while the derivation is being constructed and which are enforced off-line when the derivation is complete.

### 3.1 Robustness (the *discard* operator)

The LFG c-structures generated by *root* and *frontier* are exactly the set of fragments generated for Tree-DOP. However, they suffer from reduced compositionality due to the constraints imposed by their associated f-structures, resulting in robustness problems. This issue can be addressed by further generalising fragments generated via *root* and *frontier* by relaxing combinations of the constraints embodied in the f-structures. A third operator, *discard*, extracts fragments from f-structures by allowing attribute-value pairs to be deleted while keeping the associated c-structures and $\phi$-links constant, as shown in Figure 8. *Discard* is subject to one restriction, namely that pairs whose values are $\phi$-linked to remaining c-structure nodes are not deleted.

### 3.2 Earlier parameter estimation for LFG-DOP

There exist two suggestions for parameter estimation for LFG-DOP *simple relative-frequency (simple-RP)* and *discounted relative-frequency (discounted-RF)*. Discounted-RF is a variant of simple-RF, especially adapted for avoiding the problem of estimating *discard* generated fragments as we explain next.

Let $|f|$ be the number of occurrences of fragment $f$ in the corpus and CS a competition set containing $f$. According to simple-RF, the probability assigned to $f$ is given by the relative frequency estimate (in the multi-set of all possible fragments):

$$P(f) = \frac{|f|}{\sum_{f_x : f_x \in CS} |f_x|}$$

The probabilities of LFG-DOP fragments can be calculated in terms of their relative frequencies among all fragments generated from the corpus.

However this parameter-estimation method, known as *simple-RF*, is as biased as the relative-frequency estimation for Tree-DOP. Furthermore, this method does not account for the fact that those fragments produced by *discard* are generalisations of fragments generated by *root* and *frontier*, and may be used in the analysis of ill-formed input. Because exponentially many fragments can be generated by applying discard to each fragment produced via root and frontier, the discard-generated fragments absorb much of the probability mass under the simple-RF estimator, just like larger subtrees absorb a large probability mass in Tree-DOP.

A second parameter estimation method, *discounted-RF*, treats fragments produced by root and frontier as seen events and those generated via discard as unseen events. The relative frequencies of seen events are discounted using the Good-Turing estimator (Good, 1953) and the discounted probability mass is given to the unseen events. As discounted-RF assigns a fixed probability mass to discard-generated fragments, the exponential number of such fragments does not adversely affect the probabilities of fragments generated by root and frontier. However, this method is as biased as the first, and it still does not account for the fact that *discard* fragments are generalisations of *root* and *frontier* fragments and may be used to analyse ill-formed input.

We think that the problem of how to employ *discard* within LFG-DOP is merely a symptom of the bias of existing estimates for LFG-DOP as a whole. Next we explain how Backoff Estimation can be applied to LFG-DOP, thereby also solving the problem with how to employ discard fragments.

## 4 BackOff Estimation for LFG-DOP

Back-off parameter estimation can be applied to LFG-DOP fragments generated by *root* and *frontier* exactly as described for Tree-DOP, using a directed acyclic graph to represent the partial order between them. A directed edge points from a fragment $\langle c_x, \phi_x, f_x \rangle$ to a pair of fragments $\langle \langle c_y, \phi_y, f_y \rangle, \langle c_z, \phi_z, f_z \rangle \rangle$ if $\langle c_y, \phi_y, f_y \rangle$ and $\langle c_z, \phi_z, f_z \rangle$ compose to give $\langle c_x, \phi_x, f_x \rangle$. Composition of these fragments involves both leftmost substitution over the c-structures and unification over the f-structures.

The production of LFG-DOP fragments via *discard* involves generating all possible f-structure fragments for each fragment produced via *root* and *frontier* while keeping c-structure and $\phi$-links constant. Therefore, the backoff relation is defined in terms of f-structure unification rather than fragment composition. For discard-generated fragments, a directed edge points from a fragment $\langle c, \phi, f \rangle$ to a pair of fragments $\langle \langle c, \phi, f_y \rangle, \langle c, \phi, f_z \rangle \rangle$ if f-structures f$_y$ and f$_z$ unify to f and f $\neq$ f$_y$ $\neq$ f$_z$:

$$\langle c, \phi, f \rangle \geq_{bfk} \langle c, \phi, \{f_y \bigcup f_z\} \rangle$$

The probability of the derivation $\langle c, \phi, \{f_y \bigcup f_z\} \rangle$ is given as follows:

$$\begin{aligned} P(\langle c, \phi, \{f_y \bigcup f_z\} \rangle | R_c) &= \\ P(c, \phi | R_c) P(\{f_y \bigcup f_z\} | c, \phi, R_c) &\approx \\ P(c, \phi | R_c) P(f_y | c, \phi) P(f_z | c, \phi, f_y) \approx& \\ P(c, \phi | R_c) P(f_y | c, \phi) P(f_z | c, \phi)& \end{aligned}$$

Thus the derivation $\langle c, \phi, \{f_y \bigcup f_z\} \rangle$ embodies an independence assumption realised by the approximation $P(f_z | c, \phi, f_y) \approx P(f_z | c, \phi)$. This approximation constitutes a backoff, hence the derivation $\langle c, \phi, \{f_y \bigcup f_z\} \rangle$ is said to be a backoff of fragment $\langle c, \phi, f \rangle$. Here, backoff is used to redistribute probability mass among discard-generated LFG-DOP fragments by transferring probability mass from complex fragments to their backoffs in a stepwise manner.

### 4.1 Discussion

Backoff Estimation of the LFG-DOP model parameters, including *discard* generated fragments, naturally addresses a major aspect of how to employ LFG-DOP. Below we discuss the observations from earlier work and explain why Backoff Estimation constitutes a suitable solution.

Experiments described in (Bod and Kaplan, 2003) suggest that, at least for small corpora over limited domains, where a parse can be produced without recourse to *discard*-generated LFG-DOP fragments, the inclusion of such fragments does not significantly improve parse accuracy. Clearly, if discarding f-structures entirely and parsing only with c-structures does not result in one or more parses, then the inclusion of *discard*-generated fragments will have no impact. Therefore, it has been suggested that *discard*-generated fragments should only be included in the parse space where the input can be parsed using c-structures only but no parse is possible over fully-instantiated $\langle c, \phi, f \rangle$ fragments.

(Way, 1999) also observes that *discard* should be used to derive fragments only where absolutely necessary. He suggests that there must be a countable number of cases – such as subject-verb agreement, relative clause agreement and movement phenomena – in which unification fails and *discard*-generated fragments should be applied. He outlines some ways in which composition via *discard* could work given such a list of cases, whereby the process is triggered by the *occurrence* of unification failure and controlled by the *type* of failure that occurred.

These observations seem to lead towards an LFG-DOP model where *discard*-generated fragments are treated as "second rate" fragments that will be used only upon failure of the other fragments to produce analyses for the input. Although based on empirical experience, this constitutes an ad hoc mechanism that might mask the symptoms but will not provide the remedy for the main problem: biased parameter estimates. We think that the same can be realized in a natural manner by Backoff Estimation. In Backoff Estimation, the parameters are structured in the backoff graph that directs the estimation algorithm for realizing a kind of *soft*, probabilistic backoff.

Indeed, and as an ad hoc simplification of Backoff Estimation, it is possible to employ the partial ordering of fragments to achieve better performance also when *discrad*-generated fragments are employed. Clearly, for Tree-DOP and for *root*- and *frontier*-generated fragments in LFG-DOP, the partial ordering of fragments is used solely for parameter estimation. For *discard*-generated LFG-DOP fragments, however, this partial ordering can be further exploited in order to motivate the phased addition of such fragments to the parse space. For example, fragments can be added in layers to the parse space, starting with the most specific (i.e. the first layer of each backoff graph) and working toward the least specific. As soon as at least one parse can be produced, no more fragments are introduced and the most probable parse is determined, thus favoring parses with more complete f-structures. This approach accounts for the fact that, where one or more parses can be produced via *discard* (but not *root* and *frontier* alone), these parses can be considered to occupy a spectrum ranging from most specific to least specific, depending on the number of attribute-value pairs that have been discarded from the fragments used to derive them. Other configurations can also be envisaged. For example, following from the proposals in (Way, 1999), the fragment space could be partitioned based on the type of simple attribute-value pairs which have been discarded from each fragment.

## 5 Conclusions

This paper shows how the parameters for the LFG-DOP model can be estimated as a highly structured space of correlated events. The Backoff Estimation procedure that was originally developed for Tree-DOP (based on Phrase-Structure annotations) turns out especially suitable for LFG-DOP. In particular, Backoff Estimation provides a solution to the problem of robust LFG-DOP parsing without resorting to ad hoc, crisp mechanisms.

Future work will address various aspects of this work. Naturally, empirical experiments with the resulting LFG-DOP models need to be conducted in order to verify the empirical value of this method. There are various specific implementations of the algorithm that need to be sorted out. Furthermore, a new Maximum-Likelihood estimation procedure should be developed that takes the backoff graph into account (as that graph expresses constraints on the eligible parameter values).

**Aknowledgements:** We thank Rens Bod for illuminating discussions on LFG-DOP and the ideas discussed in this paper.

## References

Black, E., Jelinek, F., Lafferty, J., Magerman, D., Mercer, R., and Roukos, S. (1993). Towards History-based Grammars: Using Richer Models for Probabilistic Parsing. In *Proceedings of the 31st Annual Meeting of the ACL (ACL'93)*, Columbus, Ohio.

Bod, R. (1995). *Enriching Linguistics with Statistics: Performance models of Natural Language*. PhD dissertation. ILLC dissertation series 1995-14, University of Amsterdam.

Bod, R. and Kaplan, R. (1998). 'A Probabilistic Corpus-Driven Model for Lexical Functional Analysis', In *Proceedings COLING-ACL-98*, Montreal, Canada, pages 145–151.

Bod, R. (2001). What is the minimal set of fragments that achieves maximal parse accuracy? In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL'2001)*, Toulouse, France.

Bod, R. and Kaplan, R. (2003). 'A DOP model for Lexical Functional Grammar', In *Data-Oriented Parsing*, R. Bod, R. Scha and K. Sima'an (eds.), CSLI Publications, Stanford, C.A. (in press).

Bonnema, R., Buying, P., and Scha, R. (1999). A new probability model for data oriented parsing. In Dekker, P., editor, *Proceedings of the Twelfth Amsterdam Colloquium*, pages 85–90. ILLC/Department of Philosophy, University of Amsterdam, Amsterdam.

Bresnan, J. (2001). 'Lexical Functional Syntax', Oxford: Blackwell.

Charniak, E. (2000). A maximum entropy inspired parser. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-00)*, pages 132–139, Seattle, Washington, USA.

Chelba, C. and Jelinek, F. (1998). Exploiting syntactic structure for language modeling. In Boitet, C. and Whitelock, P., editors, *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*, pages 225–231, San Francisco, California. Morgan Kaufmann Publishers.

Chen, S. and Goodman, J. (1998). *An empirical study of smoothing techniques for language modeling*. Technical Report TR-10-98, Harvard University.

Good, I. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237–264.

Johnson, M. (2002). The DOP estimation method is biased and inconsistent. *Computational Linguistics*, 28(1):71–76.

Katz, S. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing (ASSP)*, 35(3):400–401.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19:313–330.

Sima'an, K. (2002). Computational complexity of probabilistic disambiguation. *Grammars*, 5(2):125–151.

Sima'an, K. and Buratto, L. (2003). Backoff Parameter Estimation for the DOP Model. In *Proceedings of the 14th European Conference on Machine Learning (ECML'03)*, Cavtat-Dubrovnik, Croatia.

Way, A. (1999). A Hybrid Architecture for Robust MT using LFG-DOP, In *Journal of Experimental and Theoretical Artificial Intelligence (Special Issue on Memory-Based Learning)*, Taylor & Francis, London.