

Data Oriented Parsing Literature Review

Jim Rankin

November 21, 2007

1 What is DOP?

The idea of Data Oriented Parsing was introduced by Remko Scha (1990). Written in 1990, it advocates a much more empirical approach to natural language parsing than was common at that time.

1.1 The Problem

Scha notes how Chomsky makes a sharp distinction between a formal grammar that defines the set of grammatical sentences in a language and the use of that language in a community, sloppiness and mistakes included. The former is usually referred to by Chomsky as “competence”, and the latter as “performance.” Scha then questions how useful this distinction is for contemporary natural language processing systems.

He first points out how linguistic theory and practical NLP systems have converged on context free grammars as a respectable theoretical model of natural language syntax also having efficient parsing algorithms. Scha then goes on to point out some drawbacks of using context free grammars to model natural language syntax.

First, trying to account for more phenomena by adding rules to a grammar means that more and more interactions between grammar rules must be inspected as the grammar grows. This is not surprising to anyone who has attempted to construct something more than a toy grammar by hand.

Second, and perhaps more important, is ambiguity. It is very common for a context free grammar to assign several parses to a single sentence. However, a human being will often perceive only one of those parses as possible. Scha asserts that the “alternative” parses produced by the grammar are only relatively less plausible and that a person generally perceives only one or a few of the most plausible ones. In other words, it is more useful to say that a parse for a sentence is only more or less plausible than some other parse, rather than saying it is correct or incorrect.

Scha then points out that these two shortcomings of formal grammars compound each other. Adding more rules to account for more syntactic phenomena

creates even more opportunities for ambiguity. Furthermore, Scha argues that these problems follow from the fact that these grammars are “competence grammars”; ambiguity of this sort is a performance concern in Chomsky’s paradigm.

1.2 The DOP Solution

Scha then introduces Data Oriented Parsing as an approach capable of rectifying these inadequacies. He points to the success of Markov word chains for selecting the most probable sentence from the output of a speech recognizer (Bahl et al. 1983; Jelinek 1986). And he cites work assigning probabilities to grammar rules to determine the most probable parse (Derouault and Merialdo 1986; Fujisaki 1984; Fujisaki et al. 1989).

DOP seeks to incorporate larger units of structure. Scha cites previous work that view new utterances as built from fragments of previous utterances. He wants to make these ideas more formal.

Fillmore et al. (1988) refer to a fragment of a tree structure labeled with syntactic and other information as a “construction.” They also introduce the term “construction grammar” to describe the act of combining constructions to form sentences. Scha wants to combine the idea of a construction grammar with statistics for each construction’s frequency of occurrence to create a new grammar formalism. He argues that this reflects the human bias for recognizing patterns and phrases that have occurred often in past experience. It is this formalism that Scha names Data Oriented Parsing.

Scha goes on to suggest ways that Data Oriented Parsing might offer insights into human language processing. This is important to note, because most of the rest of this literature review will focus on computational DOP implementations.

2 DOP Implementation

Scha does not present an implementation for DOP, or even a detailed formalism. But he does discuss what an implementation of DOP might involve. First, there must be an efficient way to represent and access the entire corpus of “past utterances.” Second, a parse created from a few previous constructions should be preferred over one that requires many. Third, parses containing frequently occurring constructions should be preferred over parses containing constructions that are more rare.

2.1 Formalization

Rens Bod (1992) took up the task of fully formalizing and implementing DOP . He gives formal definitions of *lexical* and *non-lexical labels*, *string*, *tree*, *subtree*, and *corpus*. Lexical and non-lexical labels are defined similarly to terminal and non-terminal symbols in a parse tree generated from a context free grammar. His definitions for the other terms are similar to their common computer science

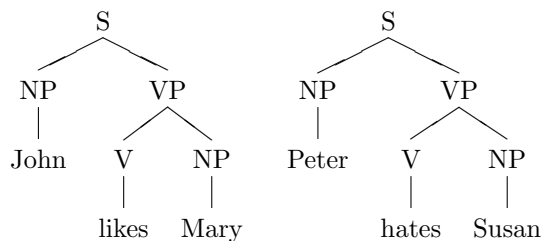


Figure 1: A DOP “corpus” (Bod 1993)

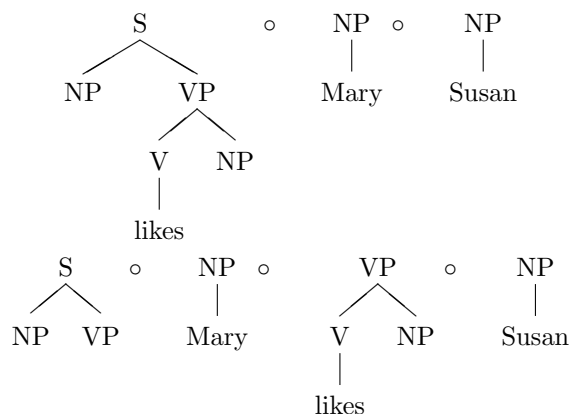


Figure 2: Ways to compose “Mary likes Susan” (Bod 1993)

usage. His definitions of *pattern* and *construction*, however, are specific to DOP and I will attempt to briefly describe them here.

A pattern is a subtree in which the children of one or more non-lexical labels have been deleted. The constructions of a tree are the union of its subtrees and patterns. If the root of a tree u is identical to the left-most non-lexical leaf of pattern t , then $t \circ u$ is the tree formed by replacing that non-lexical leaf with u . Bod calls this process *composition*. Bod then defines parsing a sentence in terms of composition that follows straight forwardly from the definitions given above. In short, a parse for a new sentence is created by composing constructions such that the yield of the resulting tree is the sentence. For example, if the “corpus” is the sentences in figure 1, the parse tree for the sentence “Mary likes Susan” can be composed from constructions as shown in figure 2. Another name for this kind of model is a Stochastic Tree Substitution Grammar (Goodman 1996).

Bod next considers probability. Most sentences will have several parses. Given a parse for a sentence, what is the conditional probability of the parse given the sentence?

Bod calculates the probability of a construction t as the ratio of the number

of times $|t|$ it appears in the corpus to the number of times any tree t' with the same root $r(t) = r(t')$ appears in the corpus.¹

$$P(t) = \frac{|t|}{\sum_{t':r(t')=r(t)} |t'|} \quad (1)$$

If $T = t_1 \circ \dots \circ t_n$ for the parse tree T of some sentence, we say $t_1 \circ \dots \circ t_n$ is a derivation of T . To calculate the probability of a derivation $t_1 \circ \dots \circ t_n$, Bod takes the product of the probabilities of all the constructions in the derivation.

$$P(t_1 \circ \dots \circ t_n) = \prod_i P(t_i) \quad (2)$$

Note that choosing this method of calculating the probability of a derivation implicitly assumes that, given its root node $r(t_i)$, each t_i is independent of every other $t_j, j \neq i$. It is common (perhaps necessary) to introduce independence assumptions when defining a generative model, but it is not clear how justified this particular assumption is. Of course, these independence assumptions are weaker than those implied by a context free grammar.

Then Bod calculates the probability of a parse as the sum of all the possible derivations of that parse. So if $D(T)$ is the set of possible derivations for parse tree T then

$$P(T) = \sum_{d \in D(T)} P(d) \quad (3)$$

To calculate the probability of a parse T_i given the sentence s Bod proposes

$$P(T_i|s) = \frac{P(T_i)}{\sum_j P(T_j)} \quad (4)$$

where T_j are all of the possible parses for s . The parse that maximizes $P(T_i|s)$ is called the *preferred parse* of s .

2.2 Implementation

Bod (1992) suggests some ideas for building a working DOP implementation. Bod (1993) introduces the first DOP implementation.

He points out that existing rule based parsing strategies can be applied to DOP. He claims that every subtree t can be seen as a production of the form $root(t) \rightarrow t$ where $root(t)$ is the root of tree t . Then new rules/productions can be applied to the non-terminals of t .

He suggests Monte Carlo techniques to avoid having to compute every possible parse for a sentence but instead estimating the preferred parse by randomly sampling the space of possible parses. The estimated probability of a parse is then the ratio of the number of times a derivation of that parse was sampled to

¹While these ideas were introduced in (Bod 1993), I'm using the equations from (Bod 2003) here as I find them to be clearer.

depth	accuracy
≤ 2	87%
≤ 3	92%
≤ 4	93%
≤ 5	93%
≤ 6	95%
≤ 7	95%
unbounded	96%

Figure 3: Bod’s reported parsing accuracy for the ATIS corpus Bod (1993)

the total number of samples. This is necessary because the number of possible derivations for a given parse is exponential in the length of the sentence. By taking a sufficiently large number of samples from the space of possible parses, it should be possible to estimate the probability of the parse with an arbitrarily small error. Specifically, the time to estimate the probability with maximal error ϵ is $O(\epsilon^{-2})$ according to Chebyshev’s inequality.²

Bod used annotated part of speech sequences from the ATIS corpus in the Penn Treebank to train and test his implementation. He trained on 675 annotated sequences and tested on 75 unannotated sequences. He performed experiments with subtrees of maximum depth 2, 3, 4, 5, 6, 7, and subtrees with unbounded depth. No timing results are reported for these experiments. He reports the results in figure 3 for these experiments, where accuracy is the percentage of sentences for which the treebank parse is identical to the parse selected by Bod’s model.

96% on this accuracy measure is an outstanding result. In fact, it still may be the highest such reported accuracy result in the literature. However, Bod’s DOP model and results have received some strong criticisms which I will now review.

²Chebyshev’s inequality places an upper bound on the probability that the absolute difference between the sample mean and the distribution mean is greater than some arbitrarily chosen value. Formally, if X_1, X_2, \dots, X_n is an iid sample, \bar{X} the sample mean, μ the distribution mean, σ the standard deviation and ϵ the proposed upper bound, Chebyshev’s inequality can be written

$$P(|\bar{X} - \mu| > \epsilon) \leq \frac{\sigma^2}{n\epsilon^2} \tag{5}$$

A consequence of this is that the number of samples needed to be confident of ϵ is $O(\epsilon^{-2})$. An interactive demonstration of Chebyshev’s inequality is available at <http://demonstrations.wolfram.com/ChebyshevsInequalityAndTheWeakLawOfLargeNumbers/> (requires free Mathematica player).

3 Criticisms

3.1 Goodman

Joshua Goodman (1996) thoroughly analyzes Bod’s work on DOP. First, he reports that other researchers have been unable to replicate Bod’s results due to insufficient details in Bod’s publications. Goodman claims to be the first to replicate the full DOP model and goes on to describe how he did this and compares his results to other models evaluated on the same data. He then discusses the specific data set Bod used to train and evaluate his implementation of the DOP model.

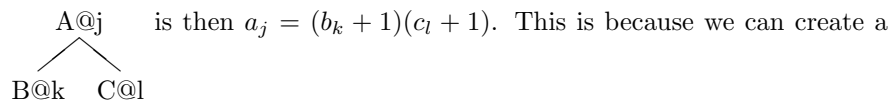
3.1.1 An equivalent PCFG

In order to replicate the DOP model, Goodman first describes a method for creating an equivalent probabilistic context free grammar (PCFG). He claims this PCFG creates the same strings and the same trees with the same probabilities as DOP. His PCFG has exactly eight rules for each node in the treebank. Thus, its size is $O(n)$ where n is the number of nodes in the treebank.

Goodman indexes every node in the treebank such that $A@k$, for example, would indicate the k th node in the treebank which happens to be labeled with non-terminal A . He also creates a new non-terminal for each node in the treebank. So A_k would denote the non-terminal created for node $A@k$.³ Goodman calls the original non-terminals like A external non-terminals and the new non-terminals like A_k internal non-terminals.

Goodman says that a PCFG subderivation is isomorphic to a DOP tree or subtree if it begins with an external non-terminal, uses internal non-terminals for intermediate steps, and ends with external non-terminals. So the DOP tree and the PCFG subderivation tree in figure 4 would be isomorphic.

Let a_j be the number of subtrees headed by $A@j$ and a the number of subtrees headed by A anywhere in the corpus. Goodman argues that the number of subtrees for the tree



subtree by choosing from any of the $(b_k + 1)$ trees on the left and any of the $(c_l + 1)$ on the right (adding 1 for the trivial subtrees consisting of only B or only C). Then for each subtree in the corpus, Goodman adds the rules in figure 5 to his PCFG.

Goodman proves that the resulting PCFG will generate derivational trees isomorphic to DOP trees having identical probabilities. I will not repeat the proof here, but hopefully the motivation for how the rule probabilities are calculated from the various subtree counts is somewhat intuitive.

³Note that $A@k$ and A_k , while having a one-to-one correspondence, are distinct. $A@k$ is a node in the treebank and A_k is a non-terminal in the PCFG that Goodman creates.

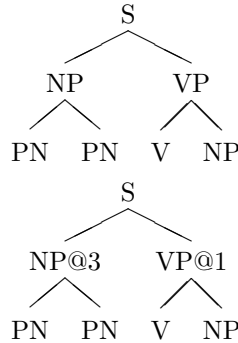


Figure 4: DOP tree and an isomorphic PCFG derivation (Goodman 1996)

$$\begin{array}{llll}
 A_j \rightarrow BC & (1/a_j) & A \rightarrow BC & (1/a) \\
 A_j \rightarrow B_k C & (b_k/a_j) & A \rightarrow B_k C & (b_k/a) \\
 A_j \rightarrow BC_l & (c_l/a_j) & A \rightarrow BC_l & (c_l/a) \\
 A_j \rightarrow B_k C_l & (b_k c_l/a_j) & A \rightarrow B_k C_l & (b_k c_l/a)
 \end{array}$$

Figure 5: Goodman’s PCFG Rules (Goodman 1996)

There is a significant difference between how Goodman selects the “best” parse compared to Bod. Goodman selects the “Maximum Constituents Parse.” This is the parse that will have the greatest number of constituents correct on average, assuming that the true tree is drawn according to the model distribution. The grammar may not even be able to produce this parse. Bod (1996) later cites this as a reason why Goodman’s model cannot truly be considered equivalent to DOP.

3.1.2 Performance analysis

Goodman’s algorithms required about 6 seconds to parse each sentence. Bod’s required about 3.5 hours per sentence, according to Bod (1995). When differences in hardware are taken into account, Goodman’s algorithm is still about 500 times faster.

Goodman describes an error in Bod’s runtime analysis of his algorithm that may explain some of the discrepancy in speed. Bod claims the asymptotic runtime of his Monte Carlo algorithm is $O(Gn^3\epsilon^{-2})$ where G is the size of the grammar, n sentence length and ϵ the maximum estimation error. However, Bod assumes the term ϵ^{-2} is constant. Because the number of potential parses for a sentence increases exponentially in the length of the sentence, the probability of the most probable parse decreases exponentially. This in turn means that the number of Monte Carlo samples needed to guarantee a maximum error of ϵ must also increase exponentially in the length of the sentence. Exponentially

increasing the number of samples will also increase the asymptotic runtime exponentially.

To assess accuracy, Goodman ran experiments using subsets of the same ATIS sentences as Bod. He found that the accuracy results for his implementation of the DOP model were not nearly as good as what Bod had reported. Compared to Bod’s 96%, for example, Goodman got 86% on a less restrictive measure. Goodman’s accuracy measures were slightly better than the results from an experiment by Pereira and Schabes (1992) run on the same ATIS data. These improvements were statistically significant for some data sets, but not others.

In order to speculate about possible reasons for the difference between his accuracy results and Bod’s, Goodman next considers Bod’s data.

3.1.3 An analysis of Bod’s Data

The ATIS data from the Penn Treebank requires some degree of “cleanup” to make it usable for parsing. Goodman ran experiments with both minimally cleaned up ATIS data and Bod’s data, which contained “more significant revisions.” The accuracy results for Bod’s data are much better than for the minimally edited data. Goodman speculates that the cleanliness of Bod’s data may be one reason for his “extraordinary results.” In fact, the only experiments that showed a statistically significant improvement over Pereira and Schabes (1992) were on Bod’s cleaned up data.

Speculating further about Bod’s data, Goodman considers the upper bound of the number of sentences that can be correctly parsed by the DOP model in principal. The DOP model can only produce the exactly correct parse of a sentence if every production in that parse occurs somewhere in the training data. Goodman does not have Bod’s data, but he does know that Bod claims that only one of his 75 test sentences had a correct parse that could not be produced from the productions in the training data.

Goodman estimates the probability that a sentence will have a production that occurs nowhere else in the training data. Because he does not have detailed knowledge of how Bod’s parser handles phenomena like ϵ -productions, unary productions and n -ary ($n > 2$) productions, Goodman produces different estimates for this value based on different assumptions. If that probability is denoted by p , the probability of selecting 75 sentences with exactly one sentence having a production that does not appear anywhere else in the data is $75p^{74}(1 - p)$. Goodman’s values for p range from 0.78 to 0.92. This implies that the probability of Bod’s test set ranges from less than 1 in a million to 1.5% in the most generous case. Goodman thus concludes that Bod made an “extraordinarily fortuitous choice of test data.”

3.2 Johnson

Johnson (2002) considers the DOP method for estimating tree probabilities. Because there had been multiple DOP estimators reported at the time of publi-

cation, Johnson makes clear that he is discussing the “DOP1” estimator, which is Bod’s DOP as described above.

Two desirable properties of a statistical estimator are that it be *consistent* and *unbiased*. The probability distributions estimated by a consistent estimator will converge on the true distribution generating the data as the number of samples increases, assuming that the true distribution and the distribution being estimated are the same. The expected value of an unbiased estimator (under the distribution generating the data) is equal to the “true” parameter value (or values) that it’s estimating.

More formally, let’s say Θ is a real valued vector space, P_θ is the probability distribution corresponding to the parameter value $\theta \in \Theta$, and $\phi(x) \in \Theta$ is a function whose input x is a vector of n samples. We call ϕ an *estimator*. For DOP1, θ would be the fragment weights, and ϕ would be the DOP1 method for calculating those weights from the corpus. If X is a vector of random variables distributed according to P_{θ^*} , $\theta^* \in \Theta$, then $\phi(X)$ is a random variable with expected value $E_{\theta^*}(\phi(X))$. The bias of ϕ at θ^* is

$$E_{\theta^*}(\phi(X)) - \theta^* \tag{6}$$

If the bias is 0, we say that ϕ is unbiased.

Now consider an estimation procedure that defines a series of estimators ϕ_n for each sample size n . We define \mathcal{L} such that $\mathcal{L}(\theta^*, \phi(x))$ is a measure of the “loss” or “cost” of using $\phi_n(x)$ to estimate θ^* . Then if $E_{\theta^*}(\mathcal{L}(\theta^*, \phi(x))) \rightarrow 0$ as $n \rightarrow \infty$, the estimation procedure is consistent. To determine whether the DOP1 estimator is consistent, Johnson uses the following “loss function.”

$$\mathcal{L}(\theta^*, \phi(x)) = \sum_{\omega \in \Omega} P_{\theta^*}(\omega)(P_{\theta^*}(\omega) - P_{\phi(x)}(\omega))^2 \tag{7}$$

Where Ω is the space of all phrase structure trees.

To demonstrate that the DOP1 estimator is inconsistent and biased, Johnson presents a simple DOP model that generates only two trees, t_1 and t_2 , with probabilities p and $1 - p$, respectively. He calculates the estimated subtree weights \hat{w} under the DOP1 model and labels the “true” weights w^* . He then calculates⁴ $P_{E(\hat{w})}(t_1) = \frac{2p}{1+p}$. Thus, since $P_{E(\hat{w})} \neq P_{w^*} = p$ (except for $p = 0$ and $p = 1$) the DOP1 estimator is biased. And because $P_{E(\hat{w})}$ does not approach $P_{E(w^*)}$ as the sample size increases, DOP1 is inconsistent.⁵

The next section will present further analysis of the consistency of DOP estimators and DOP estimators that tries to rectify the flaws reported by Johnson.

⁴I have not presented the specific trees, subtrees and associated weights used in Johnson’s calculations. But the model is simple enough that I was able to replicate the calculations with pencil and paper. The interested reader is encouraged to do so also.

⁵Although bias and consistency are usually defined in terms of parameters, Goodman (1996) explains that for DOP it is more natural to define them in terms of the distributions that the parameters specify. This is because in DOP there can be parameter vectors $\theta_1 \neq \theta_2$ such that $P_{\theta_1} = P_{\theta_2}$.

3.3 Summary of Criticisms

Here is a summary of the criticisms of DOP presented so far. Goodman levels the following criticisms at Bod’s DOP model.

1. Bod did not report enough details to replicate his results.
2. Bod underestimates the asymptotic runtime of his algorithm.
3. Bod cleaned up his data significantly before running his experiments.
4. Bod chose a surprisingly favorable dataset for testing.
5. Although Goodman believes he was able to replicate the DOP model, he was unable to replicate Bod’s reported performance.

Johnson’s criticism is the title of his paper.

1. The DOP estimator is biased and inconsistent.

4 Consistent, Efficient DOP?

Most of Goodman’s criticisms are criticisms of the experiments Bod ran and how he reported them. However, Johnson’s criticism goes more to the heart of the DOP model as consistency is generally considered a necessary property of an acceptable estimator. Is a consistent estimator for DOP possible?

4.1 Every consistent DOP estimator is biased

Prescher et al. (2004) seek to answer that question. Prescher, Scha, Sima’an and Zollman consider DOP estimators in more detail than Johnson and propose new, consistent DOP estimators they believe to be consistent.

Prescher et al. arrive at a more specific conclusion than Johnson: every unbiased estimator for DOP is inconsistent. In brief, an unbiased estimator for the DOP probability model will necessarily assign all probability mass to the full parse trees in the training corpus. Because there is no probability mass left over for trees that do not appear in the corpus, this estimator will be inconsistent. They give a detailed proof of this claim, a summary of which follows.

Let $t \in \mathcal{F}$ be the set of fragments in the corpus and $\pi(t)$ the weight assigned to fragment t , such that π induces a probability distribution on the set of fragments whose root label is $R_t = A$, i.e. $\sum_{t:R_t=A} \pi(t) = 1$. Let $f_t(d)$ be the number of times that fragment t appears in derivation d . Then the probability of d is

$$p(d) = \prod \pi(t)^{f_t(d)} \tag{8}$$

If $D(x)$ is the complete set of derivations for parse tree x then the probability of x is

$$p(x) = \sum_{d \in \mathcal{D}(x)} p(d) \quad (9)$$

Let \mathcal{X} be a countable set and $f : \mathcal{X} \rightarrow \mathcal{N}$ be a function mapping each $x \in \mathcal{X}$ to a natural number. We call x a *type*, f a *corpus* and $f(x)$ a *type frequency*. $|f| = \sum_{x \in \mathcal{X}} f(x)$ is the total count of all types in the corpus. In the case of DOP, \mathcal{X} is the set of all fragments and $f(x)$ is the number of times that fragment x appears in the treebank.

Then the unrestricted set of all probability distributions over \mathcal{X} is

$$\mathcal{M}(\mathcal{X}) = \{p : \mathcal{X} \rightarrow [0, 1] \mid \sum_{x \in \mathcal{X}} p(x) = 1\} \quad (10)$$

We can now express the DOP1 probability model as

$$\mathcal{M}_{\mathcal{DOP}} = \{p \in \mathcal{M}(\mathcal{X}) \mid \exists \pi : \forall x : p(x) = \sum_{d \in \mathcal{D}(x)} \prod_{t \in \mathcal{F}} \pi(t)^{f_t(d)}\} \quad (11)$$

and the context free grammar probability model as

$$\mathcal{M}_{\mathcal{CFG}} = \{p \in \mathcal{M}(\mathcal{X}) \mid \exists \pi : \forall x : p(x) = \prod_{r \in \mathcal{F}} \pi(r)^{f_r(x)}\} \quad (12)$$

where $r \in \mathcal{F}$ are the rules in the CFG.

The relative frequency distribution is $\tilde{p} : \mathcal{X} \rightarrow [0, 1]$ where $\tilde{p}(x) = \frac{f(x)}{|f|}$. As proven by Cover and Thomas (1991), \tilde{p} is the maximum likelihood estimate for a probability model M if and only if $\tilde{p} \in M$. Prescher et al. then demonstrate that $\tilde{p} \in M_{\mathcal{DOP}}$, but $\tilde{p} \notin M_{\mathcal{CFG}}$. This means that because the Stochastic Tree Substitution Grammar underlying DOP is more expressive than a CFG, it allows \tilde{p} as the maximum likelihood estimate which then overfits the data. Let us consider each case in turn.

Assume that $\tilde{p} \in \mathcal{M}_{\mathcal{CFG}}$. Now consider parse tree $x_0 \in \mathcal{X}$ not in the treebank, $f(x_0) = 0$. By definition, $\tilde{p}(x) = 0$ also. By the definition of $\mathcal{M}_{\mathcal{CFG}}$, $\tilde{p}(x_0) = \prod_{r \in \mathcal{F}} \pi(r)^{f_r(x_0)} = 0$. This means that $\pi(r) = 0$ for some $r \in \mathcal{F}$. However, because we only select rules that appear in the treebank for our PCFG, this is not possible. This contradiction proves our premise false and we conclude $\tilde{p} \notin \mathcal{M}_{\mathcal{CFG}}$.

For the $\mathcal{M}_{\mathcal{DOP}}$ case, we will augment the trees in our treebank such that the root node of every full parse tree is a new non-terminal symbol S . Now we define $\pi(t) = \tilde{p}(t)$ if $R_t = S$, and set $\pi(t)$ to some arbitrary value otherwise. Thus, every tree has only one derivation and $p(x) = \sum_{d \in \mathcal{D}(x)} \prod_{t \in \mathcal{F}} \pi(t)^{f_t(d)} = \sum_{d=x} \prod_{t \in \{x\}} \pi(t)^1 = \tilde{p}$. Therefore, $\tilde{p} \in \mathcal{M}_{\mathcal{DOP}}$.

Furthermore, as $\sum_{x \in \mathcal{F}} \tilde{p} = \sum_{x \in \mathcal{F}} \frac{f(x)}{|f|} = 1$, necessarily $\forall x \in \mathcal{X}, x \notin \mathcal{F} : \tilde{p}(x) = 0$. In other words, the maximum likelihood estimate will assign 0 probability to all full parse trees not in the treebank, which the authors call “complete overfitting.” A consequence of this is that every unbiased estimator will overfit the data. The authors conclude that “in designing estimators for DOP, we

should explicitly introduce bias towards predicting trees that were not observed in the corpus.” However, it is desirable that such an estimator still be consistent.

The authors propose two new estimators for DOP that are biased but consistent. The first employs held-out estimation similar to that used in n-gram language modeling. The second employs a kind of backoff smoothing.

4.2 A Held-Out Estimator for DOP: DOP*

Zollmann and Sima’an (2005) presents the DOP* Held-Out estimator in detail. The general steps are as follows. The training treebank is split into a held-out corpus (HC) and an extraction corpus (EC). Fragments are extracted from the EC then weights for those trees are calculated from the HC. The goal is to assign weights to the fragments from the EC that will maximize the joint probability of the parse trees in the HC. (Actually, it maximizes the joint probability of the HC trees that can be derived from fragments in the EC. Some parse trees in HC may not be derivable from fragments in EC.)

One way to think about maximizing the joint probability of a set of parse trees is to use Expectation Maximization with the Inside Outside algorithm. This approach is employed in Bod (2000a). However, the Inside Outside algorithm is not guaranteed to reach a global maximum so this approach can not ensure consistency.

Instead, Zollman and Sima’an make the following simplifying assumption: “maximizing the joint probability of parses in the HC is equivalent to maximizing the joint probability of their shortest derivations.” They claim the following advantages of this assumption.

- It leads to a closed form solution to the maximum likelihood estimate.
- It assigns non-zero weights to a number of fragments that is linear in the number of depth-1 fragments. There are an exponential number of fragments to consider without this assumption. This makes DOP* faster than DOP1.
- The estimator with this assumption is still consistent.
- DePauw (2000) supports the idea that DOP models that use the shortest derivation criterion perform well.

This estimator is biased, but as demonstrated by Prescher et al. (2004), every consistent DOP estimator is biased. So it makes sense to choose a consistent estimator that is biased in such a way that it has other useful properties.

Here is the DOP* algorithm for assigning weights to fragments, using the shortest derivation assumption.

1. Split the treebank into EC and HC.
2. Extract the fragments from EC.

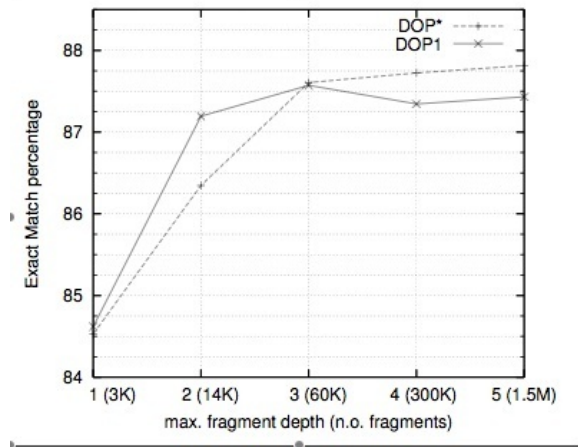


Figure 6: Performance for different maximum-depths of extracted fragments (Zollmann and Sima'an 2005)

3. Choose the shortest derivation for every tree in HC that is derivable from the fragments in EC.
4. For each fragment f_1, \dots, f_N used in the shortest derivation of some parse in HC, record the number of times it occurs, r_1, \dots, r_N in all of the shortest derivations.
5. For each f_1, \dots, f_N set $\pi(f_j) = \frac{r_j}{\sum_{k \in \{1, \dots, N\}: \text{root}(f_k) = \text{root}(f_j)} r_k}$.

This is repeated several times for different splits between EC and HC and the resulting weights $\pi(f_j)$ are averaged together (deleted estimation). Also, some probability mass, p_{unkn} , is reserved for unseen events (smoothing). p_{unkn} is defined as the relative frequency of parse trees in HC not derivable from EC fragments. p_{unkn} is distributed over depth-one fragments and fragments up to depth 3 from non-derivable parses.

The authors trained and evaluated DOP* on 10,049 annotated sentences from the OVIS corpus. OVIS is a Dutch spoken dialogue system for train table information. Accuracy results for DOP1 and DOP*, including fragments of different maximum-depths, are presented in Figure 6.

The performance of DOP1 degrades when fragments of depth greater than 3 are included. The authors speculate that this is because DOP1 neglects interdependencies of overlapping fragments. The performance of DOP* surpasses DOP1 at depth 3 and continues to improve as the maximum fragment depth increases.

Figure 7 shows parsing time in minutes for the entire corpus including fragments of different maximum depth levels. DOP* parsing time is linearly

Depth	1	2	3	4	5
DOP1	5	6	12	121	1450
DOP*	5	6	6	14	17

Figure 7: Parsing time for whole testing corpus in minutes (Zollmann and Sima’an 2005)

bounded in the depth of subtrees considered, whereas DOP1 parse time grows exponentially.

Thus, these experiments confirm that the DOP* estimator is consistent and efficient.

4.3 Backoff DOP

Prescher et al. (2004) also present a way to define backoff relations among subtrees that parallel backoff relationships for n-gram models. A tree with depth > 1 can always be expressed as the derivation of two subtrees, $t = t_1 \circ t_2$. We can then say that t_1 and t_2 participate in a backoff of t .

With this definition of backoff for subtrees, the Katz backoff for n-grams can be generalized to apply to backoff DOP estimation. Here are the steps for the Backoff DOP estimator.

1. Initialize F_{cu} as the set of all fragments in the treebank and calculate relative frequency for each $t \in F_{cu}$.
2. Discount the probability of each $t \in F_{cu}$ using, for example, leave-one-out. P_{rsv} is the the probability mass reserved for unseen events.
3. Let F_{bkf} be the set of all $t_1, t_2 \in F_{cu}$ such that $t = t_1 \circ t_2$ for some $t \in F_{cu}$. Use the Katz backoff formula to distribute P_{rsv} to each $t_i \in F_{bkf}$.
4. Let $F_{cu} = F_{bkf}$ and repeat 2 - 4 until F_{bkf} is empty.

So, in figure 8 the subtrees in the right hand column (F_{bkf}) participate in a backoff of the trees in the left hand column (F_{cu}). The trees in F_{bkf} then become F_{cu} when they appear in the left column in the next row.

A model similar to this one, BO-DOP1, was presented by Sima’an and Buratto (2003). However, Prescher et al. point out that BO-DOP1 uses DOP1 as its initialization step and thus is likely to have inherited DOP1’s inconsistency. Initializing probabilities with the relative frequencies of subtrees would eliminate this issue.

5 State of the Art DOP

Aside from issues of bias and inconsistency, there is also the question of how DOP compares to other parsing techniques in terms of performance. Bod (2003)

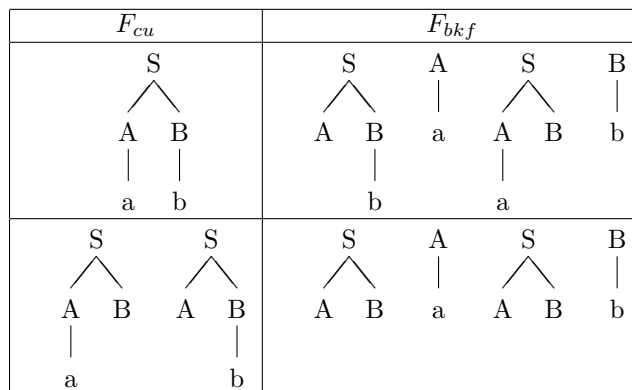


Figure 8: The trees in F_{bkf} participate in a backoff of the trees in F_{cu}

presents the best published parsing metrics scores for a DOP implementation that I have found. Charniak and Johnson (2005) cite these results as “the highest reported result for this test-set”, up until the results they were publishing.

Bod (2003) attempts to combine two different DOP models into a single model that will perform better than either alone. *Likelihood DOP* (LDOP) tries to maximize a parse tree’s likelihood, similar to the DOP1 model presented earlier. Specifically, it is the model presented in Bod (2001). Bod (2000b) introduced *Simplicity DOP* (SDOP). SDOP selects the parse tree with the derivation that uses the fewest number of subtrees.⁶ LDOP outperforms SDOP, but the parse trees chosen by LDOP and SDOP are quite different. This means that there is reason to believe that combining them into a single model will perform better than either one of them alone.

Bod implements these models using Goodman’s PCFG reduction technique presented earlier. One way that the LDOP model differs from the DOP1 model is that LDOP selects an equal number of subtrees for each possible subtree depth from 2 to 14 (in addition to all subtrees of depth 1). Bod defines α as “the number of times non-terminals of type A occur in the training data”, and modifies Goodman’s weight calculations to represent the LDOP model (figure 9). SDOP is also reduced to a PCFG where every subtree has equal probability. The resulting PCFG will always assign the highest probability to the parse with the fewest number of subtrees. These PCFG reductions yield a great improvement in processing speed. The LDOP implementation in Bod (2001) takes 220 seconds on average to parse a sentences under 100 words. The LDOP PCFG reduction averages 3.6 seconds.

Bod considers two ways to combine LDOP and SDOP. *Simplicity-Likelihood*

⁶SDOP breaks a tie, where two derivations use the same number of subtrees, by considering the frequencies of the subtrees. For every set of subtrees in the corpus having the same root, the most frequent is ranked 1, the second most frequent is ranked 2, etc. The ranks of the subtrees in each derivation are added together and the derivation with the lowest score is chosen.

$$\begin{array}{llll}
A_j \rightarrow BC & (1/a_j) & A \rightarrow BC & (1/a\alpha) \\
A_j \rightarrow B_k C & (b_k/a_j) & A \rightarrow B_k C & (b_k/a\alpha) \\
A_j \rightarrow BC_l & (c_l/a_j) & A \rightarrow BC_l & (c_l/a\alpha) \\
A_j \rightarrow B_k C_l & (b_k c_l/a_j) & A \rightarrow B_k C_l & (b_k c_l/a\alpha)
\end{array}$$

Figure 9: LDOP PCFG Rules from Bod (2003)

n	SL-DOP		LS-DOP	
	LP	LR	LP	LR
1	89.7	89.5	87.4	87.0
5	90.1	90.0	87.9	87.4
10	90.6	90.4	88.5	88.1
11	90.7	90.7	88.5	88.1
12	90.8	90.7	88.5	88.1
13	90.8	90.7	88.6	88.3
14	90.8	90.7	88.6	88.3
15	90.7	90.6	88.6	88.3
20	90.3	90.1	88.9	88.7
50	89.5	89.2	89.3	89.0
100	88.1	87.5	89.7	89.4
1,000	87.4	87.0	89.7	89.4

Figure 10: SL-DOP and LS-DOP results on the WSJ (sentences ≤ 100 words) in Bod (2003)

DOP (SL-DOP) selects the simplest parse from the n most likely parses. *Likelihood-Simplicity-DOP* (LS-DOP) selects the most likely parse from the n simplest parses.

For his experiments, this time Bod uses the “standard division of the WSJ” corpus (Marcus and Marcinkiewicz 1993); sections 2 through 21 for training, 22 for development, and 23 for testing. This makes it easier to compare Bod’s results with other published results. The ATIS corpus used in earlier experiments is not as widely used, making it more difficult to compare with other results.

The results are in table 10. LP is labeled precision and LR labeled recall. For $n = 1$, SL-DOP is equivalent to LDOP and LS-DOP is equivalent to SDOP. The best results are $LP = 90.8$ and $LR = 90.7$ for SL-DOP with $12 \leq n \leq 14$. For $n > 14$, SL-DOP converges to SDOP.

Next we will consider DOP’s relationship to other approaches to parsing.

6 Comparing DOP with Other Approaches

One of the key motivators for DOP given by Scha was using constructions from previous utterances to parse new trees. Today, this is practically taken

Vertical Order		Horizontal Order				
		$h = 0$	$h = 1$	$h \leq 2$	$h = 2$	$h = \infty$
$v = 1$	No annotation	71.27	72.5	73.46	72.96	72.62
$v \leq 2$	Sel. parents	74.75	77.42	77.77	77.50	76.91
$v = 2$	All parents	74.68	77.42	77.81	77.50	76.81
$v \leq 3$	Sel. parents	76.50	78.59	79.07	78.97	78.54
$v = 3$	All grandparents	76.74	79.18	79.74	79.07	78.72

Figure 11: F1 score for different markovizations Klein and Manning (2003)

for granted. The generic term for this is a “treebank grammar”, meaning a grammar where the rules are taken directly from the parse annotations on some corpus.

Another motivator was utilizing more context than is available in a CFG. Bod (1996) points out that a PCFG extracted from a treebank is the same as DOP using only fragments of depth 1. PCFGs are probably still the most widely used grammar formalism, but various attempts have been made to extend the PCFG model by capturing more context in PCFG rules. In this section I will compare and contrast a couple of these attempts with DOP.

My intent is not so much to compare the absolute performance of these other models to DOP’s performance, but to look at how performance changes as more context is incorporated.

6.1 Klein and Manning: Grammar Markovizations

One way to express the amount of context captured by a grammar rule is to consider its vertical and horizontal Markov orders. The vertical Markov order is the number of ancestor nodes that each child is conditioned on. The horizontal Markov order refers to the number of preceding siblings that each child is conditioned on. For example, the usual CFG or PCFG rule considers a node and all of its children. This has a vertical Markov order of 1 (the children are conditioned on only the parent node) and a horizontal Markov order that is infinite (every PCFG rule specifies every child node).

This is a somewhat arbitrary choice of Markov orders, stemming from the legacy of parsing algorithms based on context free grammars. Klein and Manning (2003) attempt to see how accurate they can make a non-lexicalized grammar. As part of this effort, they wanted to see what affect changing the vertical and horizontal Markov orders of their grammar would have on performance. Figure 11 shows the F1 score for various combinations of vertical and horizontal Markov orders. “Sel. parents” and $h \leq 2$ refers to “variable history models”, where only states meeting certain frequency minimums were included in the Markov chain.

These results do not include any smoothing. Besides changes in Markovization, the PCFG rules were not enhanced in any way. The best result is for

Depth	1	2	3	4	5	6
Score	73 ± 1	79 ± 1	80 ± 1	79 ± 1	79 ± 1	78 ± 1
Improvement	-1 ± 4	20 ± 6	23 ± 3	21 ± 4	19 ± 4	18 ± 3

Figure 12: Score shows the parse score for varying maximum depth of subtree considered in the convolution kernel. Improvement is relative reduction in error compared to the PCFG. (Collins and Duffy 2001)

$v = 3, h \leq 2$. So, vertical increases in the Markov order showed improvement for every value tried. Accuracy decreased after $h \leq 2$. Klein and Manning found that the grammar corresponding to $v \leq 2, h \leq 2$ worked best as a base onto which they could add other features. $v = 3, h \leq 2$ has a large number of states and “does not tolerate further splitting.”

6.2 Collins and Duffy: Convolution Kernels

Collins and Duffy (2001) attempt to apply kernel methods to the problem of re-ranking the parse trees output by a PCFG parser. In order to apply kernel methods to parse trees, he defines a similarity metric between trees that counts the number of common subtrees between them. They acknowledge that these are the same kind of subtrees used in DOP and cite Bod’s work.

Collins and Duffy trained a PCFG on data from the ATIS corpus and used a beam search to produce 100 candidate parse trees for each sentence. The voted perceptron algorithm was trained on 20 parses selected at random from the 100 candidate parses for each sentence. The resulting kernel was then applied to the 100 candidate parses for each test sentence and had to choose the best one.

This experiment was performed with varying maximum subtree depths and the accuracy scores are in figure 12.

Techniques like kernel methods that do not try to tell a “generative story” are generally called *discriminative* methods. DOP1 would be considered generative, for example, because it models a process of picking a subtree, then picking another subtree to compose with it repeatedly until you have a sentence. However, Prescher et al. (2004) point out that DOP is also similar to other discriminative techniques like *k-nearest neighbor* or *Memory-Based Learning* in that it tries to find trees similar to the one you want to find a derivation for. They also briefly note Bod (2003)’s Simplicity DOP and other recent non-probabilistic (and non-generative) DOP systems like DePauw (2000).

Collins (2000) also addresses the topic of parse re-ranking. Again, an initial parser produces a set of candidate parses and a second model, using additional features re-ranks them. Under this second model, the features can interact and overlap without any need to define a generative model. Some of the features selected are in figure 13. They represent various combinations of word bigram and trigrams and rules of various Markov orders. This is not the full range of possible fragments and subtrees, but it does go beyond the various Markovizations

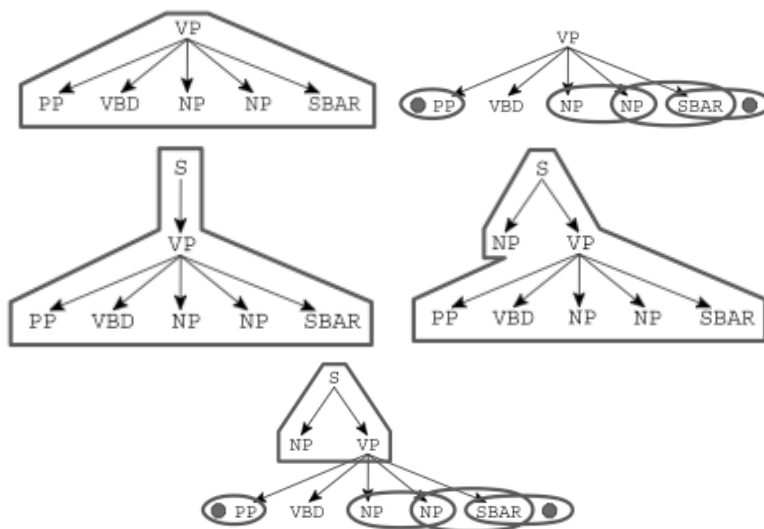


Figure 13: Various “feature windows” from Collins (2000)

tried by Klein and Manning. I chose Collins (2000) as an indicative example, but there are many other published results that use these kinds of features. For example, see Charniak and Johnson (2005).

7 Data Oriented...

There is much more research out there on Data Oriented Parsing than I was able to cover in this literature review. To find out more about these other areas, a good starting point is Rens Bod’s homepage:

<http://www.cs.st-andrews.ac.uk/rb/>

Among the issues not addressed in this literature review are Data Oriented Translation, unsupervised Data Oriented Parsing, DOP with other grammar formalisms such as HPSG, TAG and LFG, and Data Oriented Music.

8 Analysis

The models I have just presented include more context in their “rules”, “features”, or “subtrees” than a plain PCFG, but less than full-blown DOP. So what is the optimal amount of context to consider in a parsing algorithm?

The DOP answer to this question is “as much as possible.” However, this creates some problems. The exponential growth of fragment types means that it is necessary to constrain the space of allowed fragments in some way. Bod originally constrained the scope of what his model had to take into account by

Monte Carlo sampling derivations instead of calculating all of them. Even still, this took a very long time, for reasons elaborated on by Goodman. Zollman and Sima'an constrain the scope of things to consider by ignoring everything except the shortest derivation.

Models based on PCFGs come at it from the opposite direction. A PCFG makes very strong independence assumptions, which are not born out by reality. Klein and Manning showed this by varying the Markov order and finding that reducing those independence assumptions could improve accuracy.

The inconsistency of DOP1 suggests that the combinatorial number of interdependencies between fragments appears to make DOP models difficult to reason about. A PCFG's generative model is pretty straightforward. Finding a consistent, efficient generative model for DOP turned out to be more complicated than it first appeared.

The difficulty of creating a robust generative model for DOP suggests that combining DOP's "all subtrees" approach with discriminative learning techniques could yield interesting research. The techniques used by Collins (2000) work fine with overlapping features. What if the set of features used was the set of all fragments in the corpus? Would it be computationally tractable?

How can we further improve upon the parsing accuracy of DOP models? Bod (2003) reports the best results to date for a DOP parsing model. Because Zollmann and Sima'an were able to show improvement over the DOP1 model as the depth of subtrees increases, it would be interesting to compare the Zollmann and Sima'an DOP* model to the Bod (2003) model on the standard WSJ corpus data. Also, Zollmann and Sima'an only report experiments for trees up to depth 5, while Bod performed experiments using trees up to depth 14. It is likely that Zollmann and Sima'an only reported experiments up to subtree depth 5 because they were comparing with DOP1, which required 1450 minutes to parse the testing corpus. So it would also be interesting to see how DOP* would perform with subtrees of greater than depth 5.

That still leaves the question, however, of how important issues like consistency and bias are if an inconsistent, biased DOP model gets very good results. At the time of publication, the SL-DOP model had the best reported accuracy on the standard WSJ parsing task. However, it is not exactly clear why that specific model performed as well as it did. It seems to me that many of the DOP research papers appear to be trying different combinations of techniques and models and reporting what works best. But if the reasons why certain things work better than other are not clear, it is difficult for other researchers to build on that work. Certainly it is important to publish these kinds of results. But it is also important that Goodman, Johnson and Prescher et al. have been able to determine some of the mathematical properties of DOP models and suggest further improvements based on their findings.

I will end this section with some more questions that to the best of my knowledge remain unresolved.

- Has anyone repeated Klein and Manning's Markovization experiment with $v > 3$? At what point does increasing the vertical order stop paying off,

without some kind of smoothing?

- Why does accuracy stop increasing for Collins convolution tree kernel re-ranking with subtrees of depth 3? Is there a problem similar to the statistical inconsistency of DOP1 preventing it from performing better?
- Can Goodman’s PCFG reduction technique be applied to Zollman and Sima’an’s consistent, efficient estimator? How fast would the resulting system be at parsing sentences?

9 Conclusion: We’re all DOP now

At this point, we have considered both good and bad points of various DOP models and implementations. Now I would like to reflect on Remko Scha’s original conception of DOP. Here is how I summarized Scha’s ideas earlier.

“First, there must be an efficient way to represent and access the entire corpus of ‘past utterances’. Second, a parse created from a few previous constructions should be preferred over one that requires many. Third, parses containing frequently occurring constructions should be preferred over parses containing constructions that are more rare.”

The first criterion describes a treebank grammar. The third reflects how pretty much every PCFG is trained. The second one still applies more to DOP than other models, but as other models incorporate more context into their productions, they start to approach this conception as well.

So, in other words, current state of the art parsing fits pretty well into Scha’s original conception of DOP. In that sense, you could argue that Data Oriented Parsing would have been a pretty good recommendation for a research direction for a computational linguist in 1990. Specific implementations of models with DOP in the name have their strengths and weaknesses. But I think it’s good to remember that the original ideas behind DOP described are so accepted now that they are mostly taken for granted by people working on parsing today.

References

- L.B. Bahl, F. Jelinek, and R.L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2), 1983.
- R. Bod. Efficient Algorithms for Parsing the DOP Model? A Reply to Joshua Goodman. In *eprint arXiv:cmp-lg/9605031*, pages 5031–+, May 1996.
- Rens Bod. Combining semantic and syntactic structure for language modeling. In *Proceedings ICSLP-2000*, volume 3, pages 106–109, Beijing, China, 2000a.
- Rens Bod. Parsing with the shortest derivation. In *Proceedings COLING’2000*, Saarbrücken, Germany, 2000b.

- Rens Bod. What is the minimal set of subtrees that achieves maximal parse accuracy? In *Proceedings ACL'2001*, Toulouse, France, 2001.
- Rens Bod. An efficient implementation of a new dop model. In *Proceedings of the European Chapter of the Association for Computational Linguists*, 2003.
- Rens Bod. A computational model of language performance: Data oriented parsing. In *Proceedings COLING'92*, Nantes, 1992. COLING.
- Rens Bod. Using an annotated corpus as a stochastic grammar. In *Proceedings of the Sixth Conference of the European Chapter of the ACL*, 1993.
- Rens Bod. The problem of computing the most probable tree in data-oriented parsing and stochastic tree grammars. In *Proceedings of the Seventh Conference of the European Chapter of the ACL*, 1995.
- Eugene Charniak and Mark Johnson. Coarse-to fine n-best parsing and max-ent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL05)*, page 173180, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.
- Michael Collins. Discriminative reranking for natural language parsing. In *Machine Learning: Proceedings of the Seventeenth International Conference*, page 175182, Stanford, California, 2000.
- Michael Collins and Nigel Duffy. Convolution kernels for natural language. *Advances in Neural Information Processing Systems*, (NIPS 14), 2001.
- T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.
- G. DePauw. Aspects of pattern matching in dop. In *Proceedings COLING 2000*, pages 236–242, 2000.
- A.M. Derouault and B. Meriardo. Natural language modeling for phoneme-to-text transcription. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.
- C.J. Fillmore, P. Kay, and M.C. O'Connor. Regularity and idiomaticity in grammatical constructions. *Language*, 1988.
- T. Fujisaki. A stochastic approach to sentence parsing. In *Proceedings of the 10th International Conference on Computational Linguistics*, Stanford, CA, 1984.
- T. Fujisaki, F. Jelinek, J. Cocke, E. Black, and T. Nishino. A probabilistic parsing method for sentence disambiguation. In *Proc. International Parsing Workshop '89*, pages 85 – 94, Pittsburgh, Carnegie Mellon University, 1989.
- J. Goodman. Efficient algorithms for parsing the dop model. In *Proceedings Empirical Methods in Natural Language Processing*, Philadelphia, 1996.

- F. Jelinek. Self-organized language modeling for speech recognition. Technical report, IBM T.J. Watson Research Center, Yorktown Heights, N.Y., 1986.
- Mark Johnson. The dop estimation method is biased and inconsistent. *Computational Linguistics*, 28(1):71–76, 2002.
- Dan Klein and Christopher Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, July 2003.
- M. B. Santorini Marcus and M. Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2), 1993.
- Fernando Pereira and Yves Schabes. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the ACL*, pages 128–135, Newark, Delaware, 1992.
- D. Prescher, R. Scha, K. Sima'an, and A. Zollman. On the statistical consistency of dop estimators. In *Proceedings on the 14th Meeting of Computational Linguistics in the Netherlands*, Antwerp, Belgium, 2004.
- Remko Scha. Taaltheorie en taaltechnologie; competence en performance. *Computertoepassingen in de Neerlandistiek*, 1990.
- K. Sima'an and L. Buratto. Backoff parameter estimation for the dop model. In L. Todorovski H. Blockeel N. Lavrač, D. Gamberger, editor, *Proceedings of the 14th European Conference on Machine Learning*, pages 373–384. Springer, 2003.
- Andreas Zollmann and Khalil Sima'an. A consistent and efficient estimator for data-oriented parsing. *Journal of Automata, Languages and Combinatorics*, 10(2):367–388, 2005.