

# An Efficient Implementation of a New DOP Model

**Rens Bod**

ILLC, University of Amsterdam  
School of Computing, University of Leeds  
Nieuwe Achtergracht 166, NL-1018 WV Amsterdam  
rens@science.uva.nl

## Abstract

Two apparently opposing DOP models exist in the literature: one which computes the parse tree involving the most frequent subtrees from a treebank and one which computes the parse tree involving the fewest subtrees from a treebank. This paper proposes an integration of the two models which outperforms each of them separately. Together with a PCFG-reduction of DOP we obtain improved accuracy and efficiency on the Wall Street Journal treebank. Our results show an 11% relative reduction in error rate over previous models, and an average processing time of 3.6 seconds per WSJ sentence.

## 1 Introduction: A Little History

### 1.1 DOP and its *Doppelgänger*s<sup>1</sup>

The distinctive feature of the DOP approach when it was proposed in 1992 was to model sentence structures on the basis of previously observed frequencies of sentence structure fragments, without imposing any constraints on the size of these fragments. Fragments include, for instance, subtrees of depth 1 (corresponding to context-free rules) as well as entire trees.

To appreciate these innovations, it should be noted that the model was radically different from all other statistical parsing models at the time. Other models started off with a *predefined* grammar and used a corpus only for estimating the rule probabilities (as e.g. in Fujisaki et al. 1989; Black et al. 1992, 1993; Briscoe and

Waegner 1992; Pereira and Schabes 1992). The DOP model, on the other hand, was the first model (to the best of our knowledge) that proposed not to train a predefined grammar on a corpus, but to directly use corpus fragments as a grammar. This approach has now gained wide usage, as exemplified by the work of Collins (1996, 1999), Charniak (1996, 1997), Johnson (1998), Chiang (2000), and many others.

The other innovation of DOP was to take (in principle) all corpus fragments, of any size, rather than a small subset. This innovation has not become generally adopted yet: many approaches still work either with local trees, i.e. single level rules with limited means of information percolation, or with restricted fragments, as in Stochastic Tree-Adjoining Grammar (Schabes 1992; Chiang 2000) that do not include non-lexicalized fragments. However, during the last few years we can observe a shift towards using more and larger corpus fragments with fewer restrictions. While the models of Collins (1996) and Eisner (1996) restricted the fragments to the locality of head-words, later models showed the importance of including context from higher nodes in the tree (Charniak 1997; Johnson 1998a). The importance of including nonhead-words has become uncontroversial (e.g. Collins 1999; Charniak 2000; Goodman 1998). And Collins (2000) argues for "keeping track of counts of arbitrary fragments within parse trees", which has indeed been carried out in Collins and Duffy (2002) who use exactly the same set of (all) tree fragments as proposed in Bod (1992).

Thus the major innovations of DOP are:

1. the use of corpus fragments rather than grammar rules,

---

<sup>1</sup> Thanks to Ivan Sag for this pun.

2. the use of arbitrarily large fragments rather than restricted ones

Both have gained or are gaining wide usage, and are also becoming relevant for theoretical linguistics (see Bod et al. 2003a).

## 1.2 DOP1 in Retrospective

One instantiation of DOP which has received considerable interest is the model known as DOP1<sup>2</sup> (Bod 1992). DOP1 combines subtrees from a treebank by means of node-substitution and computes the probability of a tree from the normalized frequencies of the subtrees (see Section 2 for a full definition). Bod (1993) showed how standard parsing techniques can be applied to DOP1 by converting subtrees into rules. However, the problem of computing the most probable parse turns out to be NP-hard (Sima'an 1996), mainly because the same parse tree can be generated by exponentially many derivations. Many implementations of DOP1 therefore estimate the most probable parse by Monte Carlo techniques (Bod 1998; Chappelier & Rajman 2000), or by Viterbi *n*-best search (Bod 2001), or by restricting the set of subtrees (Sima'an 1999; Chappelier et al. 2002). Sima'an (1995) gave an efficient algorithm for computing the parse tree generated by the most probable derivation, which in some cases is a reasonable approximation of the most probable parse.

Goodman (1996, 1998) developed a polynomial time PCFG-reduction of DOP1 whose size is linear in the size of the training set, thus converting the exponential number of subtrees to a compact grammar. While Goodman's method does still not allow for an efficient computation of the most probable parse in DOP1, it does efficiently compute the "maximum constituents parse", i.e. the parse tree which is most likely to have the largest number of correct constituents.

Johnson (1998b, 2002) showed that DOP1's subtree estimation method is statistically biased and inconsistent. Bod (2000a) solved this problem by training the subtree probabilities by a

maximum likelihood procedure based on Expectation-Maximization. This resulted in a statistically consistent model dubbed ML-DOP. However, ML-DOP suffers from overlearning if the subtrees are trained on the same treebank trees as they are derived from. Cross-validation is needed to avoid this problem. But even with cross-validation, ML-DOP is outperformed by the much simpler DOP1 model on both the ATIS and OVIS treebanks (Bod 2000b).

Bonnema et al. (1999) observed that another problem with DOP1's subtree-estimation method is that it provides more probability to nodes with more subtrees, and therefore more probability to larger subtrees. As an alternative, Bonnema et al. (1999) propose a subtree estimator which reduces the probability of a tree by a factor of two for each non-root non-terminal it contains. Bod (2001) used an alternative technique which samples a fixed number of subtrees of each depth and which has the effect of assigning roughly equal weight to each node in the training data. Although Bod's method obtains very competitive results on the Wall Street Journal (WSJ) task, the parsing time was reported to be over 200 seconds per sentence (Bod 2003).

Collins & Duffy (2002) showed how the perceptron algorithm can be used to efficiently compute the best parse with DOP1's subtrees, reporting a 5.1% relative reduction in error rate over the model in Collins (1999) on the WSJ. Goodman (2002) furthermore showed how Bonnema et al.'s (1999) and Bod's (2001) estimators can be incorporated in his PCFG-reduction, but did not report any experiments with these reductions.

This paper presents the first published results with Goodman's PCFG-reductions of both Bonnema et al.'s (1999) and Bod's (2001) estimators on the WSJ. We show that these PCFG-reductions result in a 60 times speedup in processing time w.r.t. Bod (2001, 2003). But while Bod's estimator obtains state-of-the-art results on the WSJ, comparable to Charniak (2000) and Collins (2000), Bonnema et al.'s estimator performs worse and is comparable to Collins (1996).

In the second part of this paper, we extend our experiments with a new notion of the best parse tree. Most previous notions of best parse tree in

---

<sup>2</sup> See Bod et al. (2003b) for an overview and history of other DOP models.

DOP1 were based on a probabilistic metric, with Bod (2000b) as a notable exception, who used a simplicity metric based on the shortest derivation. We show that a combination of a probabilistic and a simplicity metric, which chooses the simplest parse from the  $n$  likeliest parses, outperforms the use of these metrics alone. Compared to Bod (2001), our results show an 11% improvement in terms of relative error reduction and a speedup which reduces the processing time from 220 to 3.6 seconds per WSJ sentence.

## 2 PCFG-Reductions of DOP

### 2.1 Formal Specification of DOP1

DOP1 parses new input by combining treebank-subtrees by means of a leftmost node-substitution operation, indicated as  $\circ$ . The probability of a parse tree is computed from the occurrence-frequencies of the subtrees in the treebank. That is, the probability of a subtree  $t$  is taken as the number of occurrences of  $t$  in the training set,  $|t|$ , divided by the total number of occurrences of all subtrees  $t'$  with the same root label as  $t$ . Let  $r(t)$  return the root label of  $t$ :

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

The probability of a derivation  $t_1 \circ \dots \circ t_n$  is computed by the product of the probabilities of its subtrees  $t_i$ :

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

An important feature of DOP1 is that there may be several derivations that generate the same parse tree. The probability of a parse tree  $T$  is the sum of the probabilities of its distinct derivations. Let  $t_{id}$  be the  $i$ -th subtree in the derivation  $d$  that produces tree  $T$ , then the probability of  $T$  is given by

$$P(T) = \sum_d \prod_i P(t_{id})$$

Thus DOP1 considers counts of subtrees of a wide range of sizes in computing the probability of a tree: everything from counts of single-level

rules to counts of entire trees. A disadvantage of this model is that an extremely large number of subtrees (and derivations) must be taken into account. Fortunately, there exists a compact PCFG-reduction of DOP1 that generates the same trees with the same probabilities, as shown by Goodman (1996, 2002). Here we will only sketch this PCFG-reduction, which is heavily based on Goodman (2002).

Goodman assigns every node in every tree a unique number which is called its address. The notation  $A@k$  denotes the node at address  $k$  where  $A$  is the nonterminal labeling that node. A new nonterminal is created for each node in the training data. This nonterminal is called  $A_k$ . Nonterminals of this form are called "interior" nonterminals, while the original nonterminals in the parse trees are called "exterior" nonterminals. Let  $a_j$  represent the number of subtrees headed by the node  $A@j$ . Let  $a$  represent the number of subtrees headed by nodes with nonterminal  $A$ , that is  $a = \sum_j a_j$ .

Goodman (1996, 2002) further illustrates this by a node  $A@j$  of the following form:

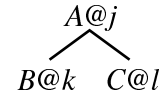


Figure 1. A node  $A@j$

To see how many subtrees it has, Goodman first considers the possibilities of the left branch. There are  $b_k$  non-trivial subtrees headed by  $B@k$ , and there is also the trivial case where the left node is simply  $B$ . Thus there are  $b_k + 1$  different possibilities on the left branch. Similarly, there are  $c_l + 1$  possibilities on the right branch. We can create a subtree by choosing any possible left subtree and any possible right subtree. Thus, there are  $a_j = (b_k + 1)(c_l + 1)$  possible subtrees headed by  $A@j$ .

Goodman then gives a simple small PCFG with the following property: for every subtree in the training corpus headed by  $A$ , the grammar will generate an isomorphic subderivation with probability  $1/a$ . Thus, rather than using the large, explicit DOP1 model, one can also use this small PCFG that generates isomorphic derivations, with identical probabilities. Goodman's construction is

as follows. For the node in figure 1, the following eight PCFG rules are generated, where the number in parentheses following a rule is its probability.

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

Figure 2. PCFG-reduction of DOP1

Goodman then shows by simple induction that subderivations headed by  $A$  with external nonterminals at the roots and leaves, internal nonterminals elsewhere have probability  $1/a$ . And subderivations headed by  $A_j$  with external nonterminals only at the leaves, internal nonterminals elsewhere, have probability  $1/a_j$  (Goodman 1996). Goodman's main theorem is that this construction produces PCFG derivations isomorphic to DOP derivations with equal probability. This means that summing up over derivations of a tree in DOP yields the same probability as summing over all the isomorphic derivations in the PCFG.

Note that Goodman's reduction method does still not allow for an efficient computation of the most probable parse tree of a sentence: there may still be exponentially many derivations generating the same tree. But Goodman shows that with his PCFG-reduction he can efficiently compute the aforementioned maximum constituents parse. Moreover, Goodman's PCFG reduction may also be used to estimate the most probable parse by Viterbi  $n$ -best search which computes the  $n$  most likely derivations and then sums up the probabilities of the derivations producing the same tree. While Bod (2001) needed to use a very large sample from the WSJ subtrees to do this, Goodman's method can do the same job with a more compact grammar.

## 2.2 PCFG-Reductions of Bod (2001) and Bonnema et al. (1999)

DOP1 has a serious bias: its subtree estimator provides more probability to nodes with more subtrees (Bonnema et al. 1999). The amount of probability given to two different training nodes

depends on how many subtrees they have, and, given that the number of subtrees is an exponential function, this means that some training nodes could easily get hundreds or thousands of times the weight of others, even if both occur exactly once. Bonnema et al. (1999) show that as a consequence too much weight is given to larger subtrees, and that the parse accuracy of DOP1 deteriorates if (very) large subtrees are included. Although this property may not be very harmful for small corpora with relatively small trees, such as the ATIS, Bonnema et al. give evidence that it leads to severe biases for larger corpora such as the WSJ. There are several ways to fix this problem. For example, Bod (2001) samples a fixed number of subtrees of each depth, which has the effect of assigning roughly equal weight to each node in the training data, and roughly exponentially less probability for larger trees (see Goodman 2002: 12). Bod reports state-of-the-art results with this method, and observes no decrease in parse accuracy when larger subtrees are included (using subtrees up to depth 14). Yet, his grammar contains more than 5 million subtrees and processing times of over 200 seconds per WSJ sentence are reported (Bod 2003).

In this paper, we will test a simple extension of Goodman's compact PCFG-reduction of DOP which has the same property as the normalization proposed in Bod (2001) in that it assigns roughly equal weight to each node in the training data. Let  $\alpha$  be the number of times nonterminals of type  $A$  occur in the training data. Then we slightly modify the PCFG-reduction in figure 2 as follows:

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

Figure 3. PCFG-reduction of Bod (2001)

We will also test the proposal by Bonnema et al. (1999) which reduces the probability of a subtree by a factor of two for each non-root nonterminal it contains. It is easy to see that this is equivalent to reducing the probability of a tree by a factor of four for each pair of nonterminals it contains,

resulting in the PCFG reduction in figure 4. Tested on the OVIS corpus, Bonnema et al.'s proposal obtains results that are comparable to Sima'an (1999) -- see Bonnema et al. (1999). This paper presents the first published results with this estimator on the WSJ.

$$\begin{array}{ll}
 A_j \rightarrow BC & (1/4) & A \rightarrow BC & (1/4\alpha) \\
 A_j \rightarrow B_k C & (1/4) & A \rightarrow B_k C & (1/4\alpha) \\
 A_j \rightarrow BC_l & (1/4) & A \rightarrow BC_l & (1/4\alpha) \\
 A_j \rightarrow B_k C_l & (1/4) & A \rightarrow B_k C_l & (1/4\alpha)
 \end{array}$$

Figure 4. PCFG-reduction of Bonnema (1999)

By using these PCFG-reductions we can thus parse with *all* subtrees in polynomial time. However, as mentioned above, efficient parsing does not necessarily mean efficient disambiguation: the exact computation of the most probable parse remains exponential. In this paper, we will estimate the most probable parse by computing the 10,000 most probable derivations by means of Viterbi  $n$ -best, from which the most likely parse is estimated by summing up the probabilities of the derivations that generate the same parse.

### 3 A New Notion of the Best Parse Tree

#### 3.1 Two Criteria for the Best Parse Tree: Likelihood vs. Simplicity

Most DOP models, such as in Bod (1993), Goodman (1996), Bonnema et al. (1997), Sima'an (2000) and Collins & Duffy (2002), use a likelihood criterion in defining the best parse tree: they take (some notion of) the most likely (i.e. most probable) tree as a candidate for the best tree of a sentence. We will refer to these models as *Likelihood-DOP* models, but in this paper we will specifically mean by "Likelihood-DOP" the PCFG-reduction of Bod (2001) given in Section 2.2.

In Bod (2000b), an alternative notion for the best parse tree was proposed based on a simplicity criterion: instead of producing the most probable tree, this model produced the tree generated by the shortest derivation with the fewest training subtrees. We will refer to this model as *Simplicity-DOP*. In case the shortest

derivation is not unique, Bod (2000b) proposes to back off to a frequency ordering of the subtrees. That is, all subtrees of each root label are assigned a rank according to their frequency in the treebank: the most frequent subtree (or subtrees) of each root label gets rank 1, the second most frequent subtree gets rank 2, etc. Next, the rank of each (shortest) derivation is computed as the sum of the ranks of the subtrees involved. The derivation with the smallest sum, or highest rank, is taken as the final best derivation producing the best parse tree in *Simplicity-DOP*.<sup>3</sup>

Although Bod (2000b) reports that *Simplicity-DOP* is outperformed by *Likelihood-DOP*, its results are still rather impressive for such a simple model. What is more important, is, that the best parse trees predicted by *Simplicity-DOP* are quite different from the best parse trees predicted by *Likelihood-DOP*. This suggests that a model which combines these two notions of best parse may boost the accuracy.

#### 3.2 Combining Likelihood-DOP and Simplicity-DOP: SL-DOP & LS-DOP

The underlying idea of combining *Likelihood-DOP* and *Simplicity-DOP* is that the parser selects the simplest tree from among the  $n$  most probable trees, where  $n$  is a free parameter. A straightforward alternative would be to select the most probable tree from among the  $n$  simplest trees. We will refer to the first combination (which selects the simplest among the  $n$  likeliest trees) as *Simplicity-Likelihood-DOP* or *SL-DOP*, and to the second combination (which selects the likeliest among the  $n$  simplest trees) as *Likelihood-Simplicity-DOP* or *LS-DOP*. Note that for  $n=1$ , *SL-DOP* is equal to *Likelihood-DOP*, since there is only one most probable tree to select from, and *LS-DOP* is equal to *Simplicity-DOP*, since there is only one simplest tree to select from. Moreover, if  $n$  gets large, *SL-DOP* converges to *Simplicity-DOP* while *LS-DOP* converges to *Likelihood-DOP*. By varying the parameter  $n$ , we will be able to compare

<sup>3</sup> As in Bod (2002), we performed one adjustment to the rank of a subtree: we averaged its rank by the ranks of all its sub-subtrees.

Likelihood-DOP, Simplicity-DOP and several instantiations of SL-DOP and LS-DOP.

### 3.3 Computational Issues

Note that Goodman's PCFG-reduction method summarized in Section 2 applies not only to Likelihood-DOP but also to Simplicity-DOP. The only thing that needs to be changed for Simplicity-DOP is that all subtrees should be assigned equal probabilities. Then the shortest derivation is equal to the most probable derivation and can be computed by standard Viterbi optimization, which can be seen as follows: if each subtree has a probability  $p$  then the probability of a derivation involving  $n$  subtrees is equal to  $p^n$ , and since  $0 < p < 1$ , the derivation with the fewest subtrees has the greatest probability.

For SL-DOP and LS-DOP, we first compute either  $n$  likeliest or  $n$  simplest trees by means of Viterbi optimization. Next, we either select the simplest tree among the  $n$  likeliest ones (for SL-DOP) or the likeliest tree among the  $n$  simplest ones (for LS-DOP). In our experiments,  $n$  will never be larger than 1,000.

## 4 Experiments

For our experiments we used the standard division of the WSJ (Marcus et al. 1993), with sections 2 through 21 for training (approx. 40,000 sentences) and section 23 for testing (2416 sentences  $\leq 100$  words); section 22 was used as development set. As usual, all trees were stripped off their semantic tags, co-reference information and quotation marks. Without loss of generality, all trees were converted to binary branching (and were reconverted to  $n$ -ary trees after parsing). We employed the same unknown (category) word model as in Bod (2001), based on statistics on word-endings, hyphenation and capitalization in combination with Good-Turing (Bod 1998: 85-87). We used "evalb"<sup>4</sup> to compute the standard PARSEVAL scores for our results (Manning & Schütze 1999). We focused on the Labeled Precision (LP) and Labeled Recall (LR) scores, as these are commonly used to rank parsing systems.

<sup>4</sup> <http://www.cs.nyu.edu/cs/projects/teufel/evalb/>

### 4.1 Comparing the PCFG-Reductions of Bod (2001) and Bonnema et al. (1999)

Our first experimental goal was to compare the two PCFG-reductions in Section 2.2, which we will refer to resp. as Bod01 and Bon99. Table 1 gives the results of these experiments and compares them with some other statistical parsers (resp. Collins 1996, Charniak 1997, Collins 1999 and Charniak 2000).

| Parser           | LP   | LR   |
|------------------|------|------|
| $\leq 40$ words  |      |      |
| Coll96           | 86.3 | 85.8 |
| Char97           | 87.4 | 87.5 |
| Coll99           | 88.7 | 88.5 |
| Char00           | 90.1 | 90.1 |
| Bod01            | 90.3 | 90.1 |
| Bon99            | 86.7 | 86.0 |
| $\leq 100$ words |      |      |
| Coll96           | 85.7 | 85.3 |
| Char97           | 86.6 | 86.7 |
| Coll99           | 88.3 | 88.1 |
| Char00           | 89.5 | 89.6 |
| Bod01            | 89.7 | 89.5 |
| Bon99            | 86.2 | 85.6 |

Table 1. Bod (2001) and Bonnema et al. (1999) compared to other parsers

While the PCFG reduction of Bod (2001) obtains state-of-the-art results on the WSJ, comparable to Charniak (2000), Bonnema et al.'s estimator performs worse and is comparable to Collins (1996). As to the processing time, the PCFG reduction parses each sentence ( $\leq 100$  words) in 3.6 seconds average, while the parser in Bod (2001, 2003), which uses over 5 million subtrees, is reported to take about 220 seconds per sentence. This corresponds to a speedup of over 60 times. It should be mentioned that the best precision and recall scores reported in Bod (2001) are slightly better than the ones reported here (the difference is only 0.2% for sentences  $\leq 100$  words). This may be explained by the fact our best results in Bod (2001) were obtained by testing various subtree restrictions until the highest accuracy was obtained, while in the current experiment we used all subtrees as given by the PCFG-reduction. In the following section

we will see that our new definition of best parse tree also outperforms the best results obtained in Bod (2001).

## 4.2 Comparing SL-DOP and LS-DOP

As our second experimental goal, we compared the models SL-DOP and LS-DOP explained in Section 3.2. Recall that for  $n=1$ , SL-DOP is equal to the PCFG-reduction of Bod (2001) (which we also called Likelihood-DOP) while LS-DOP is equal to Simplicity-DOP. Table 2 shows the results for sentences  $\leq 100$  words for various values of  $n$ .

| $n$   | 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    | <b>90.8</b> | <b>90.7</b> | 88.5        | 88.1        |
| 13    | <b>90.8</b> | <b>90.7</b> | 88.6        | 88.3        |
| 14    | <b>90.8</b> | <b>90.7</b> | 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        | <b>89.7</b> | <b>89.4</b> |
| 1,000 | 87.4        | 87.0        | <b>89.7</b> | <b>89.4</b> |

Table 2. Results of SL-DOP and LS-DOP on the WSJ (sentences  $\leq 100$  words)

Note that there is an increase in accuracy for both SL-DOP and LS-DOP if the value of  $n$  increases from 1 to 12. But while the accuracy of SL-DOP decreases after  $n=14$  and converges to Simplicity-DOP, the accuracy of LS-DOP continues to increase and converges to Likelihood-DOP. The highest accuracy is obtained by SL-DOP at  $12 \leq n \leq 14$ : an LP of 90.8% and an LR of 90.7%. This is roughly an 11% relative reduction in error rate over Charniak (2000) and Bod's PCFG-reduction reported in Table 1. Compared to the reranking technique in Collins (2000), who obtained an LP of 89.9% and an LR of 89.6%, our results show a 9% relative error rate reduction.

While SL-DOP and LS-DOP have been compared before in Bod (2002), especially in the context of musical parsing, this paper presents the

first results of SL-DOP and LS-DOP with a compact PCFG-reduction.

## 5 Conclusion

The DOP approach is based on two distinctive features: (1) the use of corpus fragments rather than grammar rules, and (2) the use of arbitrarily large fragments rather than restricted ones. While the first feature has been generally adopted in statistical NLP, the second feature has for a long time been a serious bottleneck, as it results in exponential processing time when the most probable parse tree is computed.

This paper showed that a PCFG-reduction of DOP in combination with a new notion of the best parse tree results in fast processing times and very competitive accuracy on the Wall Street Journal treebank.

This paper also re-affirmed that the coarse-grained approach of using all subtrees from a treebank outperforms the fine-grained approach of specifically modeling lexical-syntactic dependencies (as e.g. in Collins 1999 and Charniak 2000).

## References

- Black, E., J. Lafferty and S. Roukos, 1992. Development and Evaluation of a Broad-Coverage Probabilistic Grammar of English-Language Computer Manuals. *Proceedings ACL'92*, Newark, Delaware.
- Black, E., R. Garside and G. Leech, 1993. *Statistically-Driven Computer Grammars of English: The IBM/Lancaster Approach*. Rodopi: Amsterdam-Atlanta.
- Bod, R. 1992. Data Oriented Parsing, *Proceedings COLING'92*, Nantes, France.
- Bod, R. 1993. Using an Annotated Language Corpus as a Virtual Stochastic Grammar, *Proceedings AAAI'93*, Washington D.C.
- Bod, R. 1998. *Beyond Grammar: An Experience-Based Theory of Language*, Stanford, CSLI Publications.
- Bod, R. 2000a. Combining Semantic and Syntactic Structure for Language Modeling in Speech Recognition. *Proceedings ICSLP'2000*, Beijing, China.
- Bod, R. 2000b. Parsing with the Shortest Derivation, *Proceedings COLING'2000*, Saarbrücken, Germany.
- Bod, R. 2001. What is the Minimal Set of Subtrees that Achieves Maximal Parse Accuracy? *Proceedings ACL'2001*, Toulouse, France.

- Bod, R. 2002. A Unified Model of Structural Organization in Language and Music. *Journal of Artificial Intelligence Research* 17, 289-308.
- Bod, R. 2003. Do All Fragments Count? *Natural Language Engineering*, 9(2). (in press)
- Bod, R., J. Hay and S. Jannedy (eds.), 2003a. *Probabilistic Linguistics*. Cambridge, the MIT Press.
- Bod, R., R. Scha and K. Sima'an (eds.), 2003b. *Data-Oriented Parsing*. CSLI Publications.
- Bonnema, R., R. Bod, and R. Scha, 1997. A DOP Model for Semantic Interpretation, *Proceedings ACL/EACL-97*, Madrid, Spain.
- Bonnema, R., P. Buying and R. Scha, 1999. A New Probability Model for Data-Oriented Parsing, *Proceedings of the 12th Amsterdam Colloquium*, Amsterdam, The Netherlands.
- Briscoe, T. and N. Waegner, 1992. Robust Stochastic Parsing Using the Inside-Outside Algorithm. in *AAAI Workshop Notes on Statistically-Based Techniques in Natural Language Processing*.
- Chappelier J. and M. Rajman, 2000. Monte Carlo Sampling for NP-hard Maximization Problems in the Framework of Weighted Parsing. in *NLP 2000, Lecture Notes in Artificial Intelligence 1835*, 106-117.
- Chappelier J., M. Rajman and A. Rozenknop, 2002. Polynomial Tree Substitution Grammars: Characterization and New Examples. *Proceedings of 7th conference on Formal Grammar*.
- Charniak, E. 1996. Tree-bank Grammars, *Proceedings AAAI'96*, Menlo Park, Ca.
- Charniak, E. 1997. Statistical Parsing with a Context-Free Grammar and Word Statistics, *Proceedings AAAI-97*, Menlo Park, Ca.
- Charniak, E. 2000. A Maximum-Entropy-Inspired Parser. *Proceedings ANLP-NAACL'2000*, Seattle, Washington.
- Chiang, D. 2000. Statistical parsing with an automatically extracted tree adjoining grammar. *Proceedings ACL'2000*, Hong Kong, China.
- Collins, M. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies, *Proceedings ACL'96*, Santa Cruz, Ca.
- Collins, M. 1997. Three Generative Lexicalised Models for Statistical Parsing, *Proceedings ACL'97*, Madrid, Spain.
- Collins, M. 1999. *Head-Driven Statistical Models for Natural Language Parsing*, PhD thesis, University of Pennsylvania, PA.
- Collins, M. 2000. Discriminative Reranking for Natural Language Parsing, *Proceedings ICML-2000*, Stanford, Ca.
- Collins M. and N. Duffy, 2002. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. *Proceedings ACL'2002*, Philadelphia, PA.
- Eisner, J. 1996. Three new probabilistic models for dependency parsing: an exploration. *Proceedings COLING-96*, Copenhagen, Denmark.
- Eisner, J. 1997. Bilexical Grammars and a Cubic-Time Probabilistic Parser. *Proceedings Fifth International Workshop on Parsing Technologies*, Boston, Mass.
- Fujisaki, T., F. Jelinek, J. Cocke, E. Black and T. Nishino, 1989. A Probabilistic Method for Sentence Disambiguation. *Proceedings 1st Int. Workshop on Parsing Technologies*, Pittsburgh, PA.
- Goodman, J. 1996. Efficient Algorithms for Parsing the DOP Model, *Proceedings Empirical Methods in Natural Language Processing*, Philadelphia, PA.
- Goodman, J. 1998. *Parsing Inside-Out*, Ph.D. thesis, Harvard University, Mass.
- Goodman, J. 2002. Efficient Parsing of DOP with PCFG-Reductions. To appear in Bod, R., Sima'an, K. and Scha, R. (eds), *Data Oriented Parsing*, Stanford, CSLI Publications. <[www.research.microsoft.com/~joshuago/dop-csli.ps](http://www.research.microsoft.com/~joshuago/dop-csli.ps)>
- Johnson, M. 1998a. PCFG Models of Linguistic Tree Representations, *Computational Linguistics* 24(4), 613-632.
- Johnson, M. 1998b. The DOP Estimation Method is Biased and Inconsistent. Draft of Johnson (2002).
- Johnson, M. 2002. The DOP Estimation Method is Biased and Inconsistent. *Computational Linguistics*, 28, 71-76.
- Manning C. and H. Schütze, 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge.
- Marcus, M., B. Santorini and M. Marcinkiewicz, 1993. Building a Large Annotated Corpus of English: the Penn Treebank, *Computational Linguistics* 19(2).
- Pereira, F. and Y. Schabes, 1992. Inside-Outside Reestimation from Partially Bracketed Corpora. *Proceedings ACL'92*, Newark, Delaware.
- Scha, R. 1990. Taaltheorie en Taaltechnologie; Competence en Performance, in Q.A.M. de Kort and G.L.J. Leerdam (eds), *Computertoepassingen in de Neerlandistiek*, Almere: Landelijke Vereniging van Neerlandici (LVVN-jaarboek).
- Schabes, Y. 1992. Stochastic Lexicalized Tree-Adjoining Grammars. *Proceedings COLING'92*, Nantes, France.
- Sima'an, K. 1995. An optimized algorithm for Data Oriented Parsing, *Proceedings International Conference on Recent Advances in Natural Language Processing*, Tzigov Chark, Bulgaria.
- Sima'an, K. 1996. Computational Complexity of Probabilistic Disambiguation by means of Tree Grammars, *In Proceedings COLING-96*, Copenhagen, Denmark.
- Sima'an, K. 1999. *Learning Efficient Disambiguation*. PhD thesis, University of Amsterdam, The Netherlands.
- Sima'an, K. 2000. Tree-gram Parsing: Lexical Dependencies and Structural Relations, *Proceedings ACL'2000*, Hong Kong, China.