

Anssi Yli-Jyrä

Two Bracketing Schemes for the Penn Treebank

Abstract

The trees in the Penn Treebank have a standard representation that involves complete balanced bracketing. In this article, an alternative for this standard representation of the tree bank is proposed. The proposed representation for the trees is loss-less, but it reduces the total number of brackets by 28%. This is possible by omitting the redundant pairs of special brackets that encode initial and final embedding, using a technique proposed by Krauwer and des Tombe (1981). In terms of the paired brackets, the maximum nesting depth in sentences decreases by 78%. The 99.9% coverage is achieved with only five non-top levels of paired brackets. The observed shallowness of the reduced bracketing suggests that finite-state based methods for parsing and searching could be a feasible option for tree bank processing.

1. Introduction

In this article, we describe a quantitative experiment on performance limitations in syntactic complexity. In the experiment, we encode the trees in the Penn Treebank using an alternative bracketing scheme and measure the depth of the resulting structures. Our experiment reveals that the phrase structures in the Penn Treebank are actually much shallower than what one would expect. In particular, with a better bracketing scheme, there is a steep decay in the frequency of deep structures.

The steep decay in the frequency of deep bracketing is a piece of good news for finite-state based modelling of syntax. Finite-state models of syntax have been used in two ways: either as superset approximations or as subset approximations. The latter behave like context-free grammars up to a pre-defined depth of balanced bracketing. Although the size of a deterministic finite automaton implementing such an approximation grows exponentially according to the depth of bracketing, the size problem can be solved algorithmically (Yli-Jyrä 2005). The steep frequency distribution contributes, however, to an elegant motivation that we need for settling

A Man of Measure

Festschrift in Honour of Fred Karlsson, pp. 472–479

some depth limit without an arbitrary choice. According to our observations, the magical number seven (Miller 1956) can be related to such a limit in syntactic complexity.

The current result suggests that a better bracketing scheme (Krauwer & des Tombe 1981, Yli-Jyrä 2005) is a well-grounded option for finite-state models of trees in computational linguistics. It is, thus, conceivable that finite-state models could be used for grammar induction, accurate parsing, and tree pattern matching in tree banks.

2. Methods and test material

2.1 Reduced bracketing

Reduced bracketing (RB) is a linear representation for trees that optimizes the standard bracketing for easier processing. At least four approaches are known:

- (i) A stack of closing brackets can be replaced with a **super-parenthesis** symbol. For example, in the user's interface of InterLisp in 1970's, the list “((a b) (c (d)))” could be written with “((a b) (b (c)))” using this short-hand notation.
- (ii) Some left or right recursion can be marked with a special, **iterative phrase boundary**. This lossy encoding transforms structures “⟨⟨A⟨B⟩⟩C⟨D⟨E⟩⟩” and “⟨⟨A⟨B⟩⟩C⟨D⟩⟨E⟩” into “⟨⟨A/B⟩C/D/E⟩” or to “⟨A⟨B⟩/C/D/E⟩”. This encoding does not define exact interpretation without help of additional markup. It has been used earlier in a flavor of finite-state syntax advocated by Koskenniemi (1990).
- (iii) Krauwer and des Tombe (1981) proposed **condensed labelled bracketing** that can be defined as follows. Special brackets (here we use angle brackets) mark those initial and final branches that allow an omission of a bracket on one side in their realized markup. The omission is possible on the side where a normal bracket (square bracket) indicates, as a side-effect, the boundary of the phrase covered by the branch. For example, bracketing “[[A B] [C [D]]]” can be replaced with “[A B] ⟨C ⟨D⟩” using this approach.

- (iv) Johnson (1996) presented an approach that used five different kinds of brackets: “[“, “]”, “⟨⟨”, “⟨”, “⟩”. As in (iii), any number of non-square brackets can be closed with a square bracket. Additionally, any number of simple right angle brackets “⟩” can be closed with “⟨⟨”. The approach was defined for context-free grammars with binary productions only.

The first approach is less general than the third approach. For example, a super-parenthesis necessarily closes all open parentheses: the structure “(a (b(c)) (d)) (e)” can be rewritten as “(a (b(c)) (d] (e]”, but not as “(a (b (c] (d] (e]”. Furthermore, there is no opening super-parenthesis in Lisp: “(a (b(c)) (d))” cannot be rewritten as “(a [b(c] (d))”.

The second approach involves even bigger problems than the first approach: It is somewhat unclear which phrase boundaries are actually iterative. Moreover, the optimized encoding that uses iterative phrase boundaries cannot be uniquely decoded: an iterative phrase boundary occurs in coordination constructs as well as in subordinated initial or final embedding.

The third approach is essentially the same as **reduced bracketing**, studied recently in the context of finite-state grammars by the author (Yli-Jyrä 2005). It can encode adequately even extended context-free productions. New grammar frameworks (Bracketing Context-Free Grammar (BCFG), Flat BCFG and regular approximations of these) can be used to generate strings with reduced bracketing, and they can be obtained canonically from extended context-free grammars (Yli-Jyrä 2005).

The fourth approach adds little to the third one. While the third approach encodes a structure as “[A ⟨ [C] D]” the fourth approach saves one level of square brackets by encoding the same structure as “[A ⟨⟨ C ⟩ D]”.

In this article, we will adopt the **third approach**.

2.2 The Penn Treebank

The test corpus used in the experiment was the Penn Treebank from the University of Pennsylvania. The Penn Treebank is a structurally annotated corpus that consists of a sequence of sentences taken from the *Wall Street Journal*. The corpus is currently the largest widely available tree bank. The

author had an access to a version the Penn Treebank that contains altogether 1 796 379 English sentences.

Each sentence in the Penn Treebank has been annotated for part-of-speech labels and phrase-structures such as shown in Figure 1. In addition to the primary phrasal structure, some co-references and ellipses (indicating traces or shared subjects, for example) have been annotated. We studied only the context-free back-bone of the structural analyses.

```
(S1 (S (ADVP (RB Instead ))
      ( , )
      (NP-SBJ-PLE (PRP it ))
      (VP (AUX is )
          (ADVP (RB widely ))
          (VP (VBN assumed )
              (NP (-NONE- *-3 ))
              (SBAR (IN that )
                  (S (NP-SBJ (NN income-tax ) (NNS cuts ))
                    (VP (MD must )
                        (VP (AUX be )
                            (VP (ADVP (RB wholly ))
                                (VBN financed )
                                (NP (-NONE- *-4 ))
                                (PP (IN by )
                                    (NP-LGS (NP (DT some ) (NN combination ))
                                        (PP (IN of )
                                            (NP (NP (JJR higher ) (JJ indirect ) (NNS taxes ))
                                                (CC and )
                                                (NP (NP (NNS cuts ))
                                                    (PP-LOC (IN in ) (NP (JJ public ) (NN expenditure ))))))))))))))))
                (. .)))
```

Figure 1. A sample of the standard bracketing used in the Penn Treebank

3. The results of the experiment

3.1 The standard bracketing scheme

We calculated that the Penn Treebank contains 168 274 314 left or right brackets. In the trees, the maximum depth of bracketed structures was as high as 49. The frequency distribution of various nesting depths of standard bracketing (SB) is shown in Figure 2.

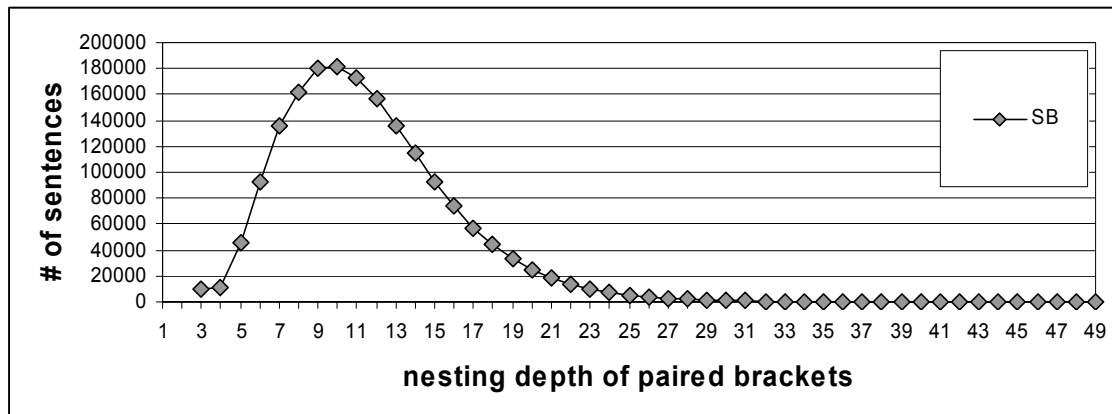


Figure 2. The nesting depth of the standard bracketing

It would be interesting to try to fit the observed distribution of bracketing depths with a statistical distribution but the space in this study does not allow this. Numerical modelling of the data is a possible subject for further study.

3.2 The reduced bracketing scheme

In our experiment, we converted the standard Penn Treebank annotation into a reduced bracketing scheme (Yli-Jyrä 2005). The transformation was carried out using a simple Perl script that is available from the author. The idea of the transformation algorithm is as follows:

- Each tree is traversed in the depth-first order from top-down, starting from the *fully bracketed* top node.
- At every node, an appropriate bracketing type (by default: *fully bracketed*) for every daughter node is assigned as follows:
 - The rightmost daughter of a fully bracketed node will be rendered with an *omitted right bracket*. If there are further daughters, the leftmost daughter will be rendered with an *omitted left bracket*.

- If a node is rendered with an omitted left bracket, then the leftmost daughter node is rendered with an *omitted left bracket*. If a node is rendered with an omitted right bracket, then the rightmost daughter node is rendered with an *omitted right bracket*.

We rendered the normal brackets as square brackets and the one-sided brackets (those with an omitted pair) as angle brackets. In total, 46 937 650 brackets (28%) were omitted, leaving 74 398 978 square brackets, 11 343 474 right-angle brackets and 35 594 212 left-angle brackets. Figure 3 shows the conversion result obtained from the sentence shown in Figure 1.

```
[S1 <S [ADVP <RB Instead ]
[ , , ]
[NP-SBJ-PLE <PRP it ]
[VP is >AUX
[ADVP <RB widely ]
<VP [VBN assumed ]
[NP <-NONE- *-3 ]
<SBAR [IN that ]
<S [NP-SBJ income-tax >NN <NNS cuts ]
<VP [MD must ]
<VP [AUX be ]
<VP [ADVP <RB wholly ]
[VBN financed ]
[NP <-NONE- *-4 ]
<PP [IN by ]
<NP-LGS [NP some >DT <NN combination ]
<PP [IN of ]
<NP [NP higher >JJR [JJ indirect ] <NNS taxes ]
[CC and ]
<NP [NP <NNS cuts ]
<PP-LOC [IN in ] <NP [JJ public ] <NN expenditure ]
< . . ]
```

Figure 3. A sample of reduced bracketing

Compared to the baseline, the maximum nesting depth in sentences decreased by 78% when we used reduced bracketing. Figure 4 compares the distributions of both kinds of bracketing depths using a logarithmic frequency scale. For reduced bracketing, the figure shows a steep decrease in the frequency of highly complex cases. The 99.9% corpus coverage was achieved with only 5 non-top levels and the 100% coverage was achieved with 10 non-top levels. The result is incidental with the famous number 7 ± 2 that characterizes many performance properties observed in psychology (Miller 1956).

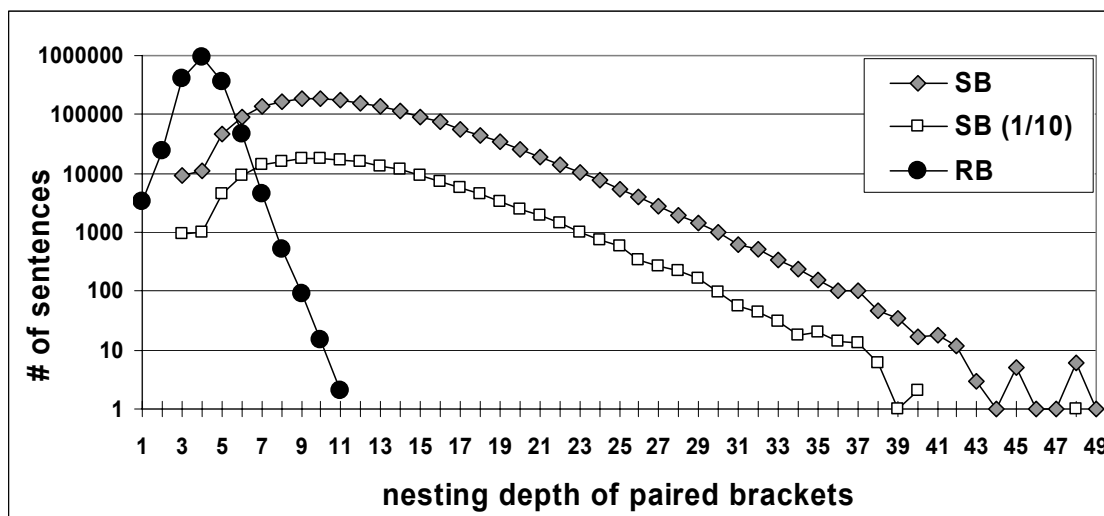


Figure 4. Distribution of bracketing depths for two bracketing schemes

4. Discussion

The reduced bracketing scheme has important consequences. For example, the observed shallowness of the resulting bracketing suggests that a finite-state based approach (Yli-Jyrä 2005) for parsing and searching tree banks could be a feasible option. Furthermore, linguistic studies on deeply embedded structures can now be focused on a portion of the corpus that represents more strikingly the abnormal cases.

In addition to the frequencies computed from the whole tree bank, Figure 4 shows the distribution of the standard bracketing depth in a random 10% sample of the tree bank. Increasing the corpus size to 100% seems to introduce more high-depth classes, but their relative frequencies remain nevertheless marginal and random. This results into slight distortion in the otherwise beautiful distribution that indicates a logarithmic decrease in the sentence frequency of high depths.

Linguistic generalizations such as the existence of a self-embedding in a competence grammar cannot be ruled out as a theoretical possibility, if a competence grammar could do without an adequate account of frequency distribution. Nevertheless, modelling the frequency distribution of different structures is crucial for most linguistic tasks, including even the task of language acquisition.

It would be tantalizing to see whether reduced bracketing could be used to adapt pure finite-state parsers to the tasks where probabilistic context-free parsers, over-generating regular approximations for context

free grammars, or fixed-point-extended finite-state models have been used earlier. A pure finite-state parser could be more efficient and it could facilitate more precise grammar learning methods by combining string patterns and an extended domain of locality in description of bracketed trees.

The observed frequency distribution indicates that the ability to process arbitrary depths of reduced bracketing makes only a negligible contribution to the performance of natural language processing. Meanwhile, the experiment suggests that reduced bracketing with limited depth is a well-grounded option for obtaining efficient techniques for syntactic analysis in computational linguistics.

References

- Johnson, Mark (1990) *Left Corner Transforms and Finite State Approximations*. Technical Report MLTT-026, Rank Xerox Research Centre, Grenoble.
- Koskenniemi, Kimmo (1990) Finite-state parsing and disambiguation. In *Papers Presented to the 13th International Conference on Computational Linguistics*, Volume 2, pp. 229–232. University of Helsinki. International Committee on Computational Linguistics.
- Krauwier, Steven & Louis des Tombe (1981) Transducers and grammars as theories of language. *Theoretical Linguistics* 8: 173–202.
- Miller, George (1956) The magical number seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review* 63: 81–97.
- Yli-Jyrä, Anssi (2005) *Contributions to the Theory of Finite-State Based [Linguistic] Grammars*. Publications of the Department of General Linguistics 38. Helsinki: University of Helsinki.

Contact information:

Anssi Yli-Jyrä
CSC – Scientific Computing Ltd.
P.O. Box 405
FI-02101 Espoo
Anssi(dot)Yli-Jyra(at)helsinki(dot)fi
<http://www.ling.helsinki.fi/~aylijyra>