Advanced Constituency Parsing: Psycholinguistically Motivated Tree-Adjoining Grammar

Lukas Mielczarek

Abstract

In this summary we explain the motivation for the search for cognitively plausible parsing algorithms and what it means for a parser to be cognitively plausible. We briefly define the formalism of *Psycholinguistically Motivated Tree-Adjoining Grammar*, present a parsing scheme for it and explain how to extract and parse a probabilistic variant of the formalism, summarising Demberg et al. (2013).

1 Motivation

Among the parsing algorithms studied in computational linguistics, not all are cognitively plausible. Following Kallmeyer (2025), one might be interested in human-inspired language processing for the reasons described in the following.

1.1 Understanding Humans

Incorporating findings about human cognition in parsers might enable in silico studies of human language processing. This can help researchers understand cognitive difficulties that humans have when processing certain phrases and may in turn contribute to our understanding of the human cognitive system.

1.2 Downstream Applications

Certain types of tasks might be solved better by human-inspired parsers. These might be tasks that require a profound level of language comprehension, tasks that necessitate dealing with incomplete sentences and tasks that encompass the generation or assessment of naturally sounding text.

2 Cognitive Plausibility

According to Demberg et al. (2013), the relevant properties a cognitively plausible parser should posses are *prediction*, *incrementality* and *connectedness*. In the following, I will outline these properties in more detail.

2.1 Prediction

Several studies have shown that humans make predictions about following material while comprehending a sentence. Altmann and Kamide (1999) have found that humans predict upcoming tokens. There is also evidence showing that upcoming structure like subcategorisation frames of verbs (Arai and Keller, 2013) and the grammatical forms of an upcoming referent (Gotzner and Spalek, 2022) are predicted.

2.2 Incrementality

Research suggests that humans process words in a sentence one by one, following their surface order, and that they do not wait until the end of the sentence before constructing a syntactic representation for it (Demberg et al., 2013). Experimental results from Tanenhaus et al. (1995) and Konieczny (2000) show that encountering a new word triggers an update of this intermediate representation.

2.3 Connectedness

Connectedness refers to the assumption that all input words are integrated into the same connected syntactic representation in the human processing device. Demberg et al. (2013) state that this holds during incremental processing: While processing a sentence, humans do not maintain unconnected fragments in their mind. Evidence for connectedness is found in Sturt and Lombardo (2005).

3 PLTAG

Psycholinguistically Motivated Tree-Adjoining Grammar (Demberg and Keller, 2008, PLTAG) is based on the mildly context-sensitive Tree-Adjoining Grammar formalism (Joshi et al., 1975; Joshi and Schabes, 1997, TAG). TAG is thought to be powerful enough for modelling natural languages while maintaining efficient parsibility (Joshi, 1985). However, as will become apparent

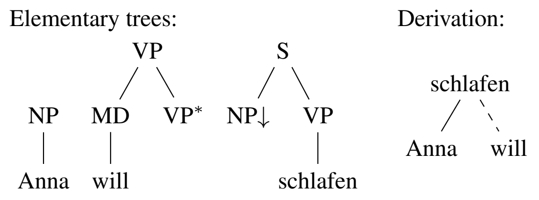

Figure 1: Elementary trees and derivation for the sentence "Anna will schlafen" (Anna wants to sleep) in TAG.

in the next section, TAG in its standard form does not respect the cognitive plausibility assumptions we have just established.

3.1 TAG is not enough

As can be seen in Figure 1, the order of derivation in regular TAG does not necessarily correspond to word order in a sentence: Either the tree for 'Anna' is substituted into 'schlafen' after processing 'will' but before integrating it into the derivation, breaking connectedness, or before processing 'will', which contradicts with incrementality. Somehow, we want 'Anna' and 'will' to connect.

3.2 Prediction and Verification

In order to allow incremental processing of a connected structure, Demberg et al. (2013) enrich TAG with rules for predicting parts of unanchored trees, so called *prediction trees*. This requires the following modifications:

Nodes now consist of a top and a bottom half. Roots only have a bottom part while substitution nodes and foot nodes only have top halves.

Indices in prediction trees show which prediction they stem from. A subscript indicates a bottom half and a superscript indicates a top half. When instantiating a prediction tree, all its markers receive a new index k not present yet in the tree. We distinguish between *canonical* elementary trees and *predictive* elementary trees.

Verification of predicted structures must at some point occur by matching with actual elementary trees. The indices disappear at this step.

Substitution and adjunction can be used to integrate a new tree into the derivation tree (*"down"*), as known from standard TAG, or to integrate the tree into a new tree (*"up"*).

3.2.1 Verification

In PLTAG, it is necessary to confirm predicted nodes using a *verification* operation by observing some canonical elementary tree τ_{verif} anchored by a succeeding word. Parts of τ_{verif} must match all node halves that were introduced by some predictive tree, identified by an index i. Possible new structure in τ_{verif} is added to the derived tree. The conditions for matching are described more rigorously in the following, based on Kallmeyer (2025)'s summary of Demberg et al. (2013).

Let l be a mapping from node half to label, t a mapping from a node half to its type (top/bottom), $\lhd$ the tree dominance relation and $\prec$ precedence.

Definition 3.1 (Correspondence)**.** A mapping f of node halves with index i in a derived tree τ to node halves in the verification tree τ_{verif} is called *correspondence* if it preserves $l, t, \lhd$, $\prec$ and is injective.

Not all correspondences are useful for PLTAG. We only use *admissible* correspondences.

Definition 3.2 (Admissible Correspondence)**.** A correspondence f is called *admissible* if it covers a contiguous subtree in the derived tree, includes its first k leaves for some k and includes its root.

Definition 3.3 (Verification)**.** We can apply a *verification for an index* i given a derived tree τ and a verification tree τ_{verif} if there is an admissible correspondence f from the node halves marked with i in τ to node halves in τ_{verif}.

If for some node u in τ_{verif}, which matches a bottom node half h in τ, only the first k children's top node halves are matched, then the subtrees rooted in the remaining children $k+1, \ldots$ are added as children of h. The index i is removed from all node halves in τ.

Since a derivation in PLTAG is incremental, the first i leaves of the derived tree are labelled with $w_1, \ldots, w_i$ for some i after a sequence of derivation steps. These derived trees are called *prefix trees*.

Definition 3.4 (Complete Derivation)**.** A derivation for a sentence $w_1, \ldots, w_n$ is called *complete* if we have scanned all leaves, the prefix tree contains no remaining substitution nodes, foot nodes or prediction markers and the label of its root is S.

4 Parsing PLTAG

The parsing scheme proposed by Demberg et al. (2013) is centered around the notion of a fringe. In Figure 2 you can find an example parse.

4.1 Fringes

The area of a prefix tree where substitions and adjunctions can happen is constrained by incrementality to the *current fringe* (cf. Figure 2).

Definition 4.1 (Node Positions)**.** For a node v we call $\langle v$ a left position and $v\rangle$ a right position.[1]

Definition 4.2 (Tree Traversal of Node Positions)**.** $npos(\tau)$ denotes a depth-first, left-to-right traversal of node positions in the tree τ.

Definition 4.3 (Fringe)**.** A *fringe* is a minimal subsequence of node positions from $npos(\tau)$ that begins to the left of the root or right of a leaf and ends to the right of the root or to the left of a leaf.

4.2 Deduction Rules

Definition 4.4 (Items)**.** The deduction *items* consist of a position i in the input sequence and $npos(\tau)$ for a prefix tree τ. We use $\bullet$ in $npos(\tau)$ to indicate the start of the current fringe.[2]

The deduction rules are defined with the goal of satisfying the incrementality condition. No new lexical items or substitution nodes can be inserted into the already scanned area preceding the current fringe. *Init* lets us start with some canonical elementary tree from the grammar and index 0. We can apply *Scan* if the curent fringe ends with a leaf node that is labelled with the next input word. As a result, the index is incremented by one. *SubstDown* can be applied if the current fringe ends with a substitution node. We select an elementary tree and substitute it into that node. *SubstUp* results in the derivation tree to be substituted into some new elementary tree. This tree is not allowed to have branches left of the spine so that no new material is inserted before the current fringe.

For adjunction, there are two down operations, *AdjDownR* and *AdjDownL* because auxiliary trees are assumed to add only nodes to the right of the spine or only to the left of it. Using *AdjDownR*, one can only adjoin at nodes whose right positions immediately follow the start of the current fringe. *AdjDownL* can only be applied to nodes whose left positions are part of the current fringe. Furthermore, *AdjUp* can be applied if the derivation tree is an auxiliary tree to adjoin it to a node in a new tree that has no left branches on its path to the root.

Lastly, *Verify* can be applied to match prediction tree indices to a new canonical elementary tree using a verification. New unmatched material is added to the derivation tree.

[1]We opted for the use of $\langle$ and $\rangle$ instead of $\bullet$ as in Kallmeyer (2025) or $^+$ and $^-$ in Demberg et al. (2013) for reasons of readability.

[2]The position of $\bullet$ is uniquely determined by i: $i = 0$ indicates the first fringe and all other values indicate the fringe starting at the ith leaf's right position.

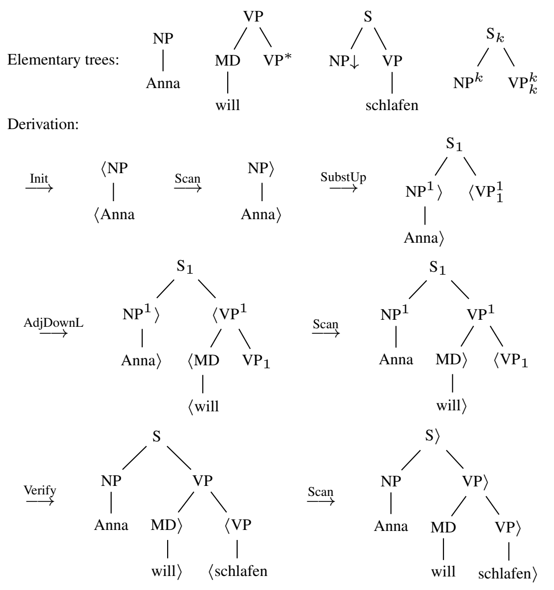

Figure 2: Parsing "Anna will schlafen" (Anna wants to sleep) in PLTAG. The current fringe is marked with $\langle, \rangle$.

5 Probabilistic PLTAG

We would like to define a probability distribution over PLTAG derivations to allow for efficient parsing. Naturally, we want the probability of a derivation to be the product of probabilities of individual derivation steps. Following Demberg et al. (2013), we define a probability model for this. For each of the following probability functions $P_X(\tau_e|...)$, it holds that $\sum_{\tau_e} P_X(\tau_e|...) = 1$.

With $P_I(\tau_e)$ we denote the probability that a derivation starts with the elementary tree τ_e. $P_S(\tau_e|\tau_d, u_e, u_d)$ is the probability of adding τ_e by substitution to the derived tree τ_d, using nodes u_e in τ_e and u_d in τ_d.[3] If u_e is a root node, τ_e is substitued into τ_d at node u_d. If u_d is a root, then τ_d is substituted into τ_e at substitution node u_e.

$P_V(\tau_v|\tau_p, u_p)$ denotes the probability of having prediction tree τ_p, that was added using node u_p, verified by a canonical tree τ_v. Finally, $P_A(\tau_e|\tau_d, u_e, u_d)$ gives us the probability of combining τ_e and τ_d using adjunction at nodes u_e and u_d. We also define $P_A(\text{NONE}|\tau_d, u_d)$ the probability of adjoining nothing to u_d. Therefore:

$$\sum_{\tau_e} P_A(\tau_e|\tau_d, u_e, u_d) + P_A(\text{NONE}|\tau_d, u_d) = 1 \tag{1}$$

[3]Since u_e is a node in τ_e and the probability therefore is trivially 0 for all other elementary trees, the intention of the original notation of Demberg et al. (2013) is unclear. It appears to be more sensible to define $P_S(\tau_e, u_e|\tau_d, u_d)$ for tuples of elementary tree τ_e and root/integration site u_e in τ_e. The same holds for P_A.

Demberg et al. (2013) make several independence assumptions to make approximating these probabilities feasible and to omit the problem of data sparsity. One idea is to predict unanchored versions of elementary trees to reduce the large number of possible items. For this, we define the *head* of a node to be the label of the daughter that is its syntactic head. Furthermore $unlex(\tau_e)$ yields τ_e without its anchor and $anchor(\tau_e)$ yields the anchor of τ_e.

Now, anchoring is simply factored out. For substitution and adjunction we have:

$$P(\tau_e|\tau_d, u_e, u_d) = P(unlex(\tau_e)|\tau_d, u_e, u_d) \\ P(anchor(\tau_e)|unlex(\tau_e), head(u_r)) \quad (2)$$

where u_r is the root of the tree that introduced u_d.

For verification we have:

$$P_V(\tau_v|\tau_p, u_p) = P(unlex(\tau_v)|\tau_p) \\ P(anchor(\tau_v)|unlex(\tau_v), head(u_p)) \quad (3)$$

where u_p is the node at which prediction tree τ_p was integrated into the already derived tree.

5.1 Supertagging

The introduction of prediction trees into TAG leads to a practical problem when parsing: Given a current fringe, there is a large number of possible prediction trees one could add and trying all of them is not feasible. To mitigate this problem, Demberg and Keller (2008) score prediction trees based on the current fringe f_d and the POS tag t_{i+1} of the next word, calling this technique *supertagging*. Then, they only use the best-scored prediction trees. With factorisation this equates to:

$$\sum_{\text{pred. tree } \tau_p} P(\tau_p|f_d, t_{i+1}) = 1 \quad (4)$$

$$P(\tau_p|f_d, t_{i+1}) = P(\tau_p|f_p, X)P(f_p, X|f_d, t_{i+1}) \quad (5)$$

where f_p is the first fringe in τ_p and X is the leaf category of the spine in τ_p.

5.2 Parsing

When parsing, Demberg et al. (2013) apply operations incrementally. They start with weight 0 for the parse items. Then, for each rule application, they add the natural logarithm of the probability of the operation to the item weight.

Furthermore, the authors use beam search. This allows for multiple parsing options to be explored which can help in cases of syntactic ambiguities.

5.3 Estimating Probabilities

For estimating the probabilities defined above, Demberg et al. (2013) use maximum likelihood estimation via counts in a treebank and smoothing. The large space of possible derived prefix trees τ_d has the consequence that most are unseen. Therefore, instead of considering τ_d as a whole, one can focus on a range of features of τ_d.

In order to observe TAG tree counts in a treebank, Demberg et al. (2013) first extract a standard LTAG from it based on top-down grammar extraction procedures. Then they gain unlexicalised prediction trees from the lexicalised elementary trees. They only extract prediction trees that are identical to some canonical elementary tree, with the exception of subtrees below and to the right of nodes on the spine. Furthermore, they make the following decisions to reduce the number of trees:

- No substitution nodes to the right of the spine are predicted.
- No unary daughters on the spine are predicted.
- Lexical co-anchors like 'either'-'or' are only predicted in rare cases.

In order to extract the required prediction trees for a decomposed treebank tree, one can compute *connection paths* between subsequent tokens (Demberg et al., 2013). These are defined as the minimal path between two leaves. Now, following a connection path from left to right, one adds a prediction tree when one encounters a node that is not part of the prefix tree or the elementary tree of the second token. This prediction tree should contain that node and all other noes on the path coming from the same elementary tree.

However, this strategy can lead to several prediction trees to add in a row. We want to restrict the number of prediction trees to add between observations of canonical trees to one for computational reasons. Therefore, Demberg et al. (2013) compile combinations of prediction trees for such cases and add them to the grammar.

On the Penn Treebank (PTB), at most five prediction trees had to be precombined (Demberg et al., 2013). In total, the number of elementary trees based on the PTB is 6700.

References

Gerry T.M Altmann and Yuki Kamide. 1999. Incremental interpretation at verbs: restricting the domain of subsequent reference. *Cognition*, 73(3):247–264.

Manabu Arai and Frank Keller. 2013. The use of verb-specific information for prediction in sentence processing. *Language and Cognitive Processes*, 28(4):525–560.

Vera Demberg and Frank Keller. 2008. A psycholinguistically motivated version of TAG. In *Proceedings of the Ninth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+9)*, pages 25–32, Tübingen, Germany. Association for Computational Linguistics.

Vera Demberg, Frank Keller, and Alexander Koller. 2013. Incremental, predictive parsing with psycholinguistically motivated Tree-Adjoining Grammar. *Computational Linguistics*, 39(4):1025–1066.

Nicole Gotzner and Katharina Spalek. 2022. Expectations about upcoming discourse referents. *International review of pragmatics : IRP*, 14(1):77 – 94.

Aravind K. Joshi. 1985. *Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions?*, Studies in Natural Language Processing, page 206–250. Cambridge University Press.

Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136–163.

Aravind K. Joshi and Yves Schabes. 1997. *Tree-Adjoining Grammars*, pages 69–123. Springer Berlin Heidelberg, Berlin, Heidelberg.

Laura Kallmeyer. 2025. Psycholinguistically motivated LTAG. Slides for the course 'Advanced Constituency Parsing' held in the winter semester 2024/25 at Heinrich Heine University Düsseldorf.

Lars Konieczny. 2000. Locality and parsing complexity. *Journal of psycholinguistic research*, 29:627–45.

Patrick Sturt and Vincenzo Lombardo. 2005. Processing coordinated structures: Incrementality and connectedness. *Cognitive science*, 29:291–305.

Michael K. Tanenhaus, Michael J. Spivey-Knowlton, Kathleen M. Eberhard, and Julie C. Sedivy. 1995. Integration of visual and linguistic information in spoken language comprehension. *Science*, 268(5217):1632–1634.