A Computational Model of Prediction in Human Parsing: Unifying Locality and Surprisal Effects

Vera Demberg (v.demberg@ed.ac.uk) and
Frank Keller (keller@inf.ed.ac.uk)
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB, UK

Abstract

There is strong evidence that human sentence processing is incremental, i.e., that structures are built word by word. Recent experiments show that the processor also predicts upcoming linguistic material on the basis of previous input. We present a computational model of human parsing that is based on a variant of tree-adjoining grammar and includes an explicit mechanism for generating and verifying predictions, while respecting incrementality and connectedness. An algorithm for deriving a lexicon from a treebank, a fully implemented parser, and a probability model for this formalism are also presented. We devise a linking function that explains processing difficulty as a combination of prefix probability (surprisal) and verification cost. The resulting model captures locality effects such as the subject/object relative clause asymmetry, as well as surprisal effects such as prediction in *either . . . or* constructions.

Keywords: Sentence Processing; Incrementality; Prediction; Surprisal; Locality Effects; Tree-adjoining Grammar.

Introduction

Evidence from psycholinguistic research suggests that language comprehension is largely *incremental*, i.e., that comprehenders build an interpretation of a sentence on a word-by-word basis. Evidence for incrementality comes from speech shadowing, self-paced reading, and eye-tracking studies (Marslen-Wilson, 1973; Konieczny, 2000; Tanenhaus et al., 1995): as soon as readers or listeners perceive a word in a sentence, they integrate it as fully as possible into a representation of the sentence thus far. They experience differential processing difficulty during this integration process, depending on the properties of the word and its relationship to the preceding context.

There is also evidence for full *connectivity* in human language processing (Sturt & Lombardo, 2005). Full connectivity means that all words are connected by a single syntactic structure; the parser builds no unconnected tree fragments, even for the incomplete sentences (sentence prefixes) that arise during incremental processing.

Furthermore, there is evidence that readers or listeners make *predictions* about upcoming material on the basis of sentence prefixes. Listeners can predict an upcoming post-verbal element, based on the semantics of the preceding verb (Kamide et al., 2003). Prediction effects can also be observed in reading. Staub & Clifton (2006) showed that following the word *either* readers predict *or* and the complement that follows it; processing was facilitated compared to structures that include *or* without *either*. In an ERP study, van Berkum et al. (1999) found that listeners use contextual information to predict specific lexical items and experience processing difficulty if the input is incompatible with the prediction.

The concepts of incrementality, connectedness, and prediction are closely related: in order to guarantee that the syntactic structure of a sentence prefix is fully connected, it may be necessary to build phrases whose lexical anchors (the words that they relate to) have not been encountered yet. Full connectedness ensures that a fully interpretable structure is available at any point during incremental sentence processing.

In this paper, we explore how these key psycholinguistic concepts (incrementality, connectedness, and prediction) can be realized within a new version of tree-adjoining grammar, which we call Psycholinguistically Motivated TAG (PLTAG). We propose a formalization of PLTAG and a linking theory that derives predictions of processing difficulty from it. We then present an implementation of this model and evaluate it against key experimental data relating to incrementality and prediction. The resulting model is shown to offer a unified framework that captures both locality effects and surprisal effects in sentence processing.

Background

Among existing models of sentence processing, two stand out as potential candidates for accounting for prediction effects. One of them is Dependency Locality Theory (DLT), proposed by Gibson (1998). A central notion in DLT is *integration cost*, a distance-based measure of the amount of processing effort required when the head of a phrase is integrated with its syntactic dependents. In other words, dependents in DLT predict the existence of a subsequent head, and the verification of these predictions causes processing cost at the head, based on its distance from the dependents.

A key experimental result captured by DLT is the fact that subject relative clauses (SRCs) as in (1a) are easier to process than object relative clauses (ORCs) as in (1b). Shorter reading times are observed on the verb *attacked* for SRCs compared to ORCs (King & Just, 1991).

(1) a. The reporter that attacked the senator admitted the error.
b. The reporter that the senator attacked admitted the error.

At the relative clause verb *attacked*, a dependency to the relative pronoun *that* is constructed; in the SRC, this involves a distance of one, while in the ORC, the subject *the senator* intervenes, resulting in a distance of two, thus explaining the higher processing cost in DLT terms.

DLT has been shown to also capture a range of other complexity results, including processing overload phenomena such as center embedding and cross-serial dependencies (Gibson, 1998). However, DLT is not a broad coverage theory: it captures the integration costs at main verbs and nouns, but makes no predictions for any other syntactic categories. This limits its usefulness in accounting for corpus data (Demberg & Keller, 2008a).

Hale (2001) proposed surprisal as an alternative measure of processing difficulty, based on ideas from probabilistic parsing. When a new word is processed during incremental interpretation, the probability of the sentence up to the new word is compared to the probability of the sentence up to the previous word. The amount of change in the probability distribution that occurs (the relative entropy of the two distributions) corresponds to the processing difficulty experienced at the new word. This means that words that are highly predictable (low relative entropy) incur low processing difficulty, while surprising words incur high processing difficulty.

As an example, consider the sentence in (2). Here, Staub & Clifton (2006) found that *an essay* is processed more quickly in (2a) than in (2b). This is captured straightforwardly by surprisal: *either ... or* is highly likely to be followed by an NP, while *or* without *either* can be followed by a wide range of phrases (including S), encountering an NP is thus more surprising in this case, resulting in elevated reading times.

(2) a. Peter read either a book or an essay in the school magazine.
b. Peter read a book or an essay in the school magazine.

Surprisal captures a range of sentence processing effects, including certain garden path effects, speed-up effects in verb-final contexts, and word order asymmetries (Hale, 2001; Levy, 2008). It is not capable, however, to account for the SRC/ORC asymmetry, as Levy (2008) shows.

DLT and surprisal therefore model complementary aspects of sentence processing. While DLT can be regarded as a backward-looking measure that focuses on integrating previous information with new information, surprisal can be seen as forward-looking, measuring whether the new input meets the comprehender's expectations. Recently, Demberg & Keller (2008a) conducted a broad-coverage evaluation of DLT and surprisal (on the Dundee Corpus, a collection of newspaper text annotated with eye-movements), and found that the predictions of the two theories are uncorrelated and account for complementary parts of the variance in the corpus reading times.

The challenge, therefore, is to develop a model of sentence processing that not only captures the properties of incrementality, connectivity, and prediction, but is also capable of explaining the complementary processing effects explained by DLT and surprisal.

Modeling Explicit Prediction

We propose a theory of sentence processing guided by the principles of incrementality, connectedness, and prediction. The core assumption of our proposal is that a sentence processor that maintains explicit predictions about the upcoming structure has to validate these predictions against the input it encounters. Using this assumption, we can naturally combine the forward-looking aspect of surprisal (sentence structures are computed incrementally and unexpected continuations cause difficulty) with the backward-looking integration view of DLT (previously predicted structures are verified against new evidence, leading to processing difficulty as predictions decay with time).

In order to build a model that implements this theory, we require an incremental parser that is capable of building fully connected structures and generating explicit predictions from which we can then derive a measure of processing difficulty. Existing parsers and grammar formalisms do not meet this specification. While there is substantial previous work on incremental parsing, none of the existing models observes full connectivity. One likely reason for this is that full connectivity cannot be achieved using canonical linguistic structures as assumed in standard grammar formalisms such as CFG, CCG, TAG, LFG, or HPSG. Instead, a stack has to be used to store partial structures and retrieve them later when it has become clear (through additional input) how to combine them.

Here, we therefore use a new variant of the tree-adjoining grammar (TAG) formalism which realizes full connectedness. The key idea is that in cases where new input cannot be combined immediately with the existing structure, we need to predict additional syntactic material, which needs to be verified against future input later on. Our variant of TAG is called Psycholinguistically Motivated TAG (PLTAG). It is outlined below and described in more detail in Demberg & Keller (2008b).

Incremental Processing with PLTAG

Tree-Adjoining Grammar

Tree-adjoining grammar (TAG) was developed by Joshi et al. (1975) as a linguistically inspired grammar formalism. It makes a fundamental distinction between initial trees and auxiliary trees. Initial trees are non-recursive and are used in substitution operations, as illustrated by the tree with the lexical anchor *the* in Figure 1a, and the trees for *senator* and *attacked* in Figure 1b. Auxiliary trees are recursive structures and are integrated into a derivation with the adjunction operation; examples are the trees with lexical anchor *or* in Figure 2d and the tree for *that* in Figure 1a. Both initial and auxiliary trees can have zero or more substitution nodes, i.e., nodes that another tree must substitute into; substitution nodes are marked with $\downarrow$. Auxiliary trees furthermore have exactly one foot node marked with *, which always has the same category as the tree's root node (rendering it recursive). Most TAG grammars are assumed to be lexicalized (LTAG); lexicalization of a grammar means that all trees have a lexical anchor, i.e., they are associated with a lexical item.

Psycholinguistically Motivated TAG

PLTAG extends normal LTAG in that it specifies not only the canonical lexicon containing lexicalized initial and auxiliary trees, but also a predictive lexicon which contains potentially unlexicalized trees, which we will call *prediction trees*. Each node in a prediction tree is annotated with indices of the form $\frac{s_j}{s_j}$, where inner nodes have two identical indices, root nodes only have a lower index and foot and substitution nodes only have an upper index. The reason for only having half of the indices is that these nodes (root, foot, and substitution nodes) still need to combine with another tree in order to build a full node. If an initial tree substitutes into a substitution node, the node where they are integrated becomes a full node, with the upper half contributed by the substitution node and the lower half contributed by the root node.

Prediction trees have the same shape as trees from the normal lexicon, with the difference that they do not contain sub-

stitution nodes to the right of their spine (the spine is the path from the root node to the anchor), and that their spine does not have to end with a lexical item. The reason for the missing right side of the spine and the missing lexical item are considerations regarding the granularity of prediction. This way, for example, we avoid predicting verbs with specific subcategorization frames (or even a specific verb) at the point of encountering the determiner of an ORC subject (as in Figure 1, discussed in more detail below). In general, we only predict upcoming structure as far as we need it, i.e., as required by connectivity or subcategorization. (However, this is a preliminary assumption, the optimal prediction grain size remains an open research question.)

PLTAG allows the same basic operations (substitution and adjunction) as normal LTAG, the only difference is that these operations can also be applied to prediction trees. In addition, we assume a verification operation, which is needed to validate previously integrated prediction trees. The tree against which verification happens has to always match the predicted tree in shape (i.e., the verification tree must contain all the nodes with a unique, identical index that were present in the prediction tree, and in the same order; any additional nodes present in the verification tree must be below the prediction tree anchor or to the right of its spine). This means that the verification operation does not introduce any tree configurations that would not be allowed by normal LTAG. (Due to space restrictions, we cannot provide a formal equivalence proof of PLTAG and LTAG here.) Note that substitution or adjunction with a predictive tree and the verification of that tree always occur pairwise, since each predicted node has to be verified. A valid parse for a sentence must not contain any nodes that are still annotated as being predictive – all of them have to be validated through verification by the end of the sentence.

In PLTAG, prediction occurs in two cases: when required by connectivity, and when required by subcategorization.

Prediction through Connectivity As briefly mentioned above, canonical elementary trees can not always be connected directly to a previously built syntactic structure. Examples are situations when two dependents precede a head, or when a grandparent and a child have been encountered, but the head of the parent node has not. This happens, for instance, at the integration of the second determiner in the ORC in (1b), as illustrated in Figure 1. The elementary tree for *the* cannot directly be combined with the preceding relative clause structure. The intervening structure will only later be provided by the trees for the noun *senator* and the verb *attacked* (see Figure 3). If we want to maintain connectivity at this point, we therefore need to predict this intervening structure (see the right hand side tree in Figure 1).[1] Predicted structures are marked with unique indices, indicating which tree they originally came from. In our example, the nodes originating from the tree structure of *senator* have index $S2$, and nodes from the tree for *attacked* have index $S1$.[2]

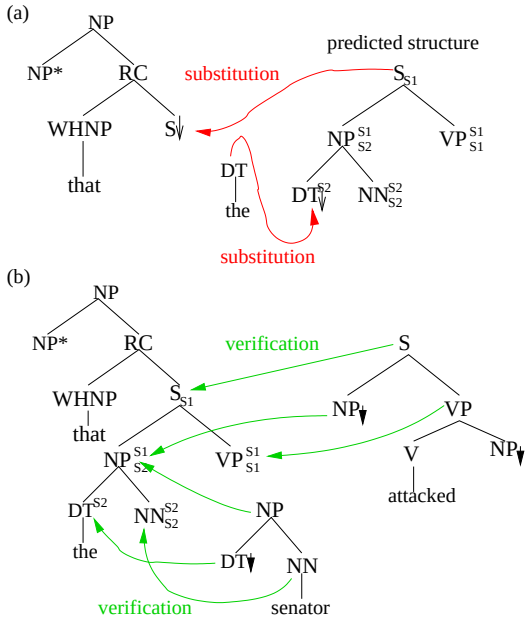


Figure 1: Prediction and Verification

Prediction through Subcategorization Another source of predictions are the lexicon entries themselves via their subcategorization frames. Subcategorization in TAG is expressed through substitution nodes, which have to be filled with an argument in order to construct a valid sentence. Each substitution node that is to the right of the tree's anchor naturally becomes a prediction during the parsing process. Modifiers are generally not predicted in our framework, unless they are needed for connectivity.[3]

We also exploit TAG's extended domain of locality in order to construct lexicon entries such that they encode lexical entries together even if they occur as two separate words. We can use this to explain predictive facilitation for *either* ... *or* and related constructions (Staub & Clifton 2006; see Background section above). For the *either* ... *or* case, we assign a lexicon entry to *either* which predicts the occurrence of the conjunction *or*, as well as predicting a coordinate structure that combines two entities of the same category, see Figure 2a.

When processing an *either* ... *or* disjunction in PLTAG, processing at *or* will be facilitated compared to a simple *or* construction. For the sequence *Peter read a book or*, the *or* occurs unexpectedly, and can be attached either at the NP level or at the S level (see Figure 2), leading to an ambiguity which will have to be resolved later on. In contrast, *or* was predicted already at *either* for the sequence *Peter read either a book or*, and will therefore be less costly to integrate: the probability of *or* given the predicted *either* structure is higher than the probability of *or* given the structure without *either*. In addition, there is no NP/S-coordination ambiguity, see Figure 2b.

[1] Because of the recursivity of natural language, it is possible that there are infinitely many ways to connect two trees. Although embedding depth can be infinite in theory, we assume that it is finite and indeed very small due to limitations of human memory.

[2] For efficiency reasons during parsing, we pre-combine prediction trees that we find in the training data, and later do not allow the parsing algorithm to integrate prediction trees which it then does not use.

[3] Whether or not modifiers are predicted syntactically is a subject for further research. Preliminary evidence suggests that modifiers are predicted when they are required by the discourse context.

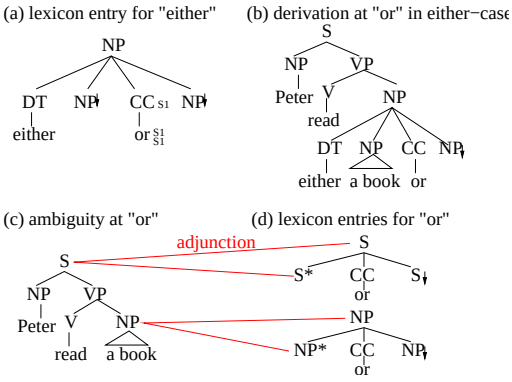


Figure 2: Extended domain of locality for expressions that trigger predictions.

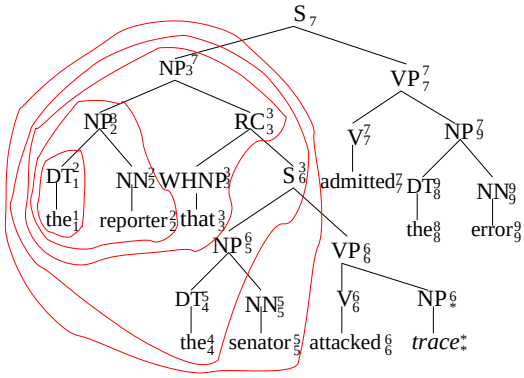


Figure 3: Generating lexicon entries from the Penn Treebank for an example sentence.

The Incremental Parser

In order to obtain a computational model that implements the prediction processes defined by PLTAG we require an incremental probabilistic parsing algorithm, which in turn requires a lexicon and a training set from which we can estimate the probabilistic model for the parser. We describe these components in turn.

Lexicon Induction and Treebank Transformation

We induce the lexicon needed for our incremental version of TAG from the Penn Treebank, complemented by noun phrase annotation (Vadas & Curran, 2007), Propbank (Palmer et al., 2003), and an adapted version of Magerman's (1994) head percolation table. These additional resources help determine the elementary trees using procedures proposed by Xia et al. (2000) and allow us to distinguish arguments (for which we generate an initial tree, and a substitution node in the parent tree) from modifiers (for which we generate auxiliary trees).

Figure 3 shows how a syntactic tree in the Treebank is cut up into elementary trees by the lexicon induction process. Each node is indexed with the number of the word that is its lexical anchor. So the tree for *the* includes the *the* node and the DT node as the tree's root node, while the tree for *that* contains all the nodes with index 3, i.e., the lexical anchor, the inner nodes WHNP and RC, the substitution node S, the root node NP and the foot node NP.

Once the trees have been segmented into elementary trees, we calculate the connection path for each prefix, as proposed by Lombardo & Sturt (2002). A connection path for words $w_1 \dots w_n$ is the minimal structure that is needed to connect all words $w_1 \dots w_n$ into the same syntactic tree. The structure needed for each of the first five words in Figure 3 is indicated by the circles enclosing the connection path at each stage.

We then use the connection paths and the canonical elementary trees to determine which parts of the structure are included in the connection path for words $w_1 \dots w_n$, but are not part of any of the elementary trees with feet $w_1 \dots w_n$. In Figure 3, this occurs for the first time at word w_4: its connection path contains nodes with indices 5 and 6. This means that parts of the structure from lexical items w_5 and w_6 have to be predicted at w_4, and two prediction trees are generated (they can be pre-combined, for the result see the predicted structure in Figure 1a).

For our lexicon induced from the Penn Treebank, we found that the average ambiguity per lexical item is 4.5 trees (the distribution is Zipfian, meaning that there are a few words with very high ambiguity, and many words with very low ambiguity, or just one lexicon entry). Among the derivations where predictive trees were needed in order to achieve connectivity, 96.15% of cases used one prediction tree, in 3.55% of cases, two prediction trees had to be combined before connectivity was achieved (as in our ORC example case discussed above); in less than 0.4% of cases were three to six trees needed, and never more than seven.

Parsing Algorithm

The parsing algorithm is strictly incremental and only allows operations that maintain the full connectivity of the partial trees that cover words $w_1 \dots w_i$. The algorithm processes alternative analyses in parallel and does not do any kind of backtracking.

When a new word w_{i+1} is encountered, the algorithm retrieves the elementary trees for this word from the lexicon, and tries to integrate each of them with previous analyses, while making sure that only correct PLTAG trees are derived at each step. The parser can also use trees from the predictive lexicon after each new word that was read in. The operations in the parsing algorithm are substitution, adjunction and verification, as outlined in the description of PLTAG above.

For illustration, assume we are parsing the sentence *The reporter that the senator attacked admitted his error.* We first retrieve trees for the lexeme *the* from the lexicon, and one of those entries is the tree for *the* shown in Figure 1. Next the algorithm predicts a number of structures that are compatible with the current structure and then reads in the next word, *reporter*. One of the trees for reporter has the structure shown in Figure 1 for the word *senator*, and the prefix tree can be integrated with the new tree by substituting the tree for *the* into the tree for *reporter* (without using any predictions). Next, prediction trees are generated and attached, and we then encounter the word *that*, retrieve its tree structure and adjoin it into the NP of the prefix tree *the senator*. When we try to attach the next round of prediction trees, one of them is the prediction tree shown at the right hand side of Figure 1a. This prediction tree is substituted into the open substitution node with category S in the prefix tree, and the tree structure for

the following word *the* can then be substituted into the open predictive substitution node with category DT. Again, more prediction trees will then be generated and integrated with the prefix, but they are not needed at encountering the next word, *senator*. Here, the first verification operation takes place. The tree for *senator* can validate all nodes with index *S*2, and also introduces the lexicalized node *senator*. Another verification takes place at the word *attacked*, and this time, all nodes with index *S*1 can be validated, and the V node and the *attacked* node, as well as the right NP substitution node, are also added to the structure.

Probability Model

The probabilities of the analyses are calculated incrementally during the parsing process. When a new tree ε is integrated into the partial derivation β, we retrieve the probability (a maximum likelihood estimate from the Penn Treebank) of ε conditional on the integration point η_β (a substitution node for the substitution operation or an adjunction node for the adjunction operation, or the prediction tree π_β it is verifying in the case of the verification operation). To reduce data sparseness, we estimate the probability of ε as its unlexicalized tree structure τ_ε multiplied with the probability of the anchor λ_ε given the unlexicalized tree structure. The integration point η_β is characterized by its category c_β, its prediction status ϕ_β (prediction tree or not) and, in the adjunction operation, its position p_β among competing adjunction sites. The anchor λ_β of a tree depends on its prediction status. It is either the lexeme *w* and the part-of-speech tag *t* for full elementary trees, or its leaf category for prediction trees. We smooth the probabilities to alleviate data sparseness problems using the smoothing algorithm proposed by Brants (2000).

(1) **Substitution:**
$\sum_\varepsilon P_s(\varepsilon|\eta_\beta) = 1$
where $P(\varepsilon|\eta_\beta) = P_s(\tau_\varepsilon|c_\beta,\phi_\beta)P(\lambda_\varepsilon|\tau_\varepsilon,\lambda_\beta)$

(2) **Adjunction:**
$\sum_\varepsilon P_a(\varepsilon|\eta_\beta) + P_a(NONE|\eta_\beta) = 1$
where $P(\varepsilon|\eta_\beta) = P_a(\tau_\varepsilon|c_\beta,p_\beta,\phi_\beta)P(\lambda_\varepsilon|\tau_\varepsilon,\lambda_\beta)$
and $P(NONE|\eta_\beta) = P_a(NONE|c_\beta,p_\beta,\phi_\beta)$

(3) **Verification:**
$\sum_\varepsilon P_v(\varepsilon|\pi_\beta) = 1$
where $P(\varepsilon|\pi_\beta) = P_v(\tau_\varepsilon|\pi_\beta)P(w_\varepsilon|\tau_\varepsilon)$

Modeling Processing Difficulty

The PLTAG formalism proposed in the previous sections is designed to implement a specific set of assumptions about human language processing (strong incrementality with full connectedness, prediction, ranked parallel processing). The formalism forms the basis for the processing theory, which uses the parser states to derive estimates of processing difficulty. We now need a linking theory that specifies the mathematical relationship between parser states and processing difficulty in our model.

During processing, the elementary tree of each new word ε_{w_i} is integrated with any previous structure ($\beta_{w_1\ldots w_{i-1}}$), and a set of syntactic expectations is generated (these expectations can be easily read off the generated tree in the form of predicted trees π). Each of these predicted trees π has a timestamp *t* that encodes when it was first predicted, or last activated (i.e., accessed). Based on the timestamp, a tree's nodes' decay *d* at verification time is calculated, under the assumption that recently-accessed structures are easier to integrate.

In our model, processing difficulty *D* is thus incurred during the construction of the syntactic analyses, as calculated from the probabilities of the elementary trees (this directly corresponds to Haleian surprisal calculated over PLTAG structures instead of over CFG structures, see the first line of Equation (4) below). In addition to this, *D* has a second component, the cost of verifying earlier predictions, which subject to a decay *d* (see the second line of Equation (4)). The overall processing difficulty *D* at word w_i is therefore:

$$(4)\quad \begin{aligned} D_{w_i} &= -\log \sum_{\beta_{1\ldots w_i}} P(\beta_{1\ldots w_i}) + \log \sum_{\beta_{1\ldots w_{i-1}}} P(\beta_{1\ldots w_{i-1}}) \\ &\quad -\log \sum_{\pi} P(\pi)^{(1-d^{t_\pi})} \end{aligned}$$

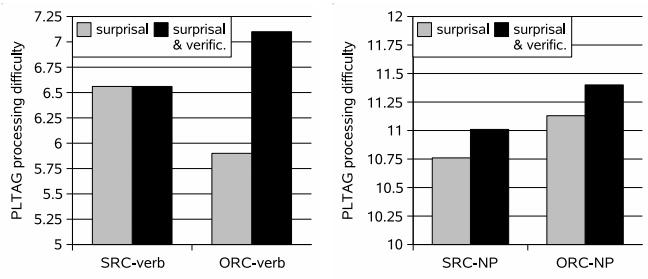
Note that the prefix probabilities $\sum_{\beta_{1\ldots w_i}} P(\beta_{1\ldots w_i})$, which are needed to calculate surprisal, fall out of the parsing process naturally, because of strict incrementality.

The verification cost component of *D* bears similarities to DLT integration costs, but we do not calculate distance in terms of number of discourse referents intervening between a dependent and its head. Rather verification cost is determined by the number of words intervening between a prediction and its verification, subject to decay. This captures the intuition that a prediction becomes less and less useful the longer ago it was made, as it decays from memory with increasing distance.

Experiments and Results

We tested our model on the SRC/ORC asymmetry and on the 48 *either ... or* sentences from Staub & Clifton's (2006) experiment. The modeling results reported here are based on a decay factor of $d = 0.8$, and the number of timesteps was set to the number of intervening words. The probabilities for the PLTAG grammar were derived from the Penn Treebank using maximum likelihood estimation.

Figure 4 shows the model predictions for the SRC and ORC sentences in (1). Model predictions and reading times are very similar for both sentences in the main clause regions, so we focus on the relative clause regions. As detailed in the Background section, experimental evidence indicates that processing time is higher at the ORC verb compared to the SRC verb. The graph in Figure 4a shows that we correctly predict this fact for the full version of the model, i.e., the version that includes both the surprisal component and the verification component in Equation (4). In a baseline model that only includes surprisal, but no verification, we incorrectly predict that the ORC verb is read faster than the SRC verb. This is consistent with Levy's (2008) observation that a probabilistic context-free grammar derived from the Penn Treebank, combined with the surprisal linking hypothesis, is unable to predict the ORC/SRC asymmetry correctly. In both version of our model, predicted reading time for the relative clause NP (see Figure 4b) is slightly higher for the ORC than the SRC. The version with surprisal and verification predicts



(a) verb region: *attacked* (b) NP region: *the senator*

Figure 4: PLTAG predictions for the verb region and the NP region for subject and object relative clauses.

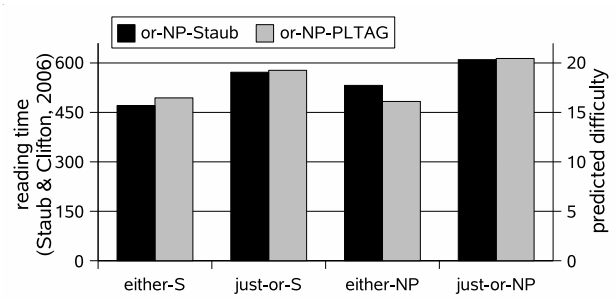


Figure 5: Mean fist pass time and mean PLTAG prediction for the post-*or* for the sentences used by Staub & Clifton (2006).

higher costs across the board, due to the fact that it has to predict and later verify a noun when integrating the determiner.

Figure 5 graphs the predictions for the full model (surprisal and verification components) for the *either* ... *or* sentences of Staub & Clifton (2006). The graph shows the first pass reading times found experimentally for the NP following *or*. Differences between the *either* and the no-*either* conditions were significant according to a paired t-test. Our model was run on the exact same sentences and replicates this pattern very well: the presence of *either* facilitates reading at the post-*or* NP in both the NP coordination and the S coordination condition. (The graph shows the model run with the same parameters as in the surprisal and verification condition in the RC experiment. A surprisal-only version of our model would predict the same pattern, but with lower difficulty predictions for the *either*-conditions.) This results demonstrate that our PLTAG model is also able to capture effects that can be explained by surprisal, but not by DLT.

Conclusions

We presented a computational model of human parsing based on PLTAG, a psycholinguistically motivated version of tree-adjoining grammar. The design of PLTAG was guided by the principles of incrementality and connectedness and includes an explicit mechanism for generating and verifying syntactic predictions. An algorithm for deriving a lexicon from a treebank, a fully implemented parser, and a probability model for PLTAG were also presented. This was complemented by a linking function that explains processing difficult as a combination of prefix probability (surprisal) and verification cost.

We demonstrated that the resulting model captures experimental results from the literature, and can explain both locality and prediction effects, which standard models of sentence processing like DLT and surprisal are unable to account for simultaneously. Our model therefore constitutes an important step towards a unified theory of human parsing. In future work, we will evaluate our model against a broader range of data, both from experiments and from eye-tracking corpora.

References

Brants, T. (2000). TnT: A statistical part-of-speech tagger. In *Proceedings of the 6th Conference on Applied Natural Language Processing*, Seattle.

Demberg, V., & Keller, F. (2008a). Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. *Cognition, 109*, 193–210.

Demberg, V., & Keller, F. (2008b). A psycholinguistically motivated version of tag. In *Proceedings of the 9th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+9)*, Tuebingen, Germany.

Gibson, E. (1998). Linguistic complexity: locality of syntactic dependencies. *Cognition 68*, (pp. 1–76).

Hale, J. (2001). A probabilistic earley parser as a psycholinguistic model. In *Proceedings of the 2nd Conference of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, PA.

Joshi, A., Levy, L., & Takahashi, M. (1975). Tree adjunct grammars. *Journal of the Computer and System Sciences, 10*.

Kamide, Y., Scheepers, C., & Altmann, G. T. (2003). Integration of syntactic and semantic information in predictive processing: Cross-linguistic evidence from German and English. *Psycholinguistic Research, 32*.

King, J., & Just, M. A. (1991). Individual differences in syntactic processing: The role of working memory. *Journal of Memory and Language, 30*, 580–602.

Konieczny, L. (2000). Locality and parsing complexity. *Journal of Psycholinguistic Research, 29*, 627–645.

Levy, R. (2008). Expectation-based syntactic comprehension. *Cognition, 106*, 1126–1177.

Lombardo, V., & Sturt, P. (2002). Incrementality and lexicalism. In *Lexical Representations in Sentence Processing, John Benjamins: Computational Psycholinguistics Series*, (pp. 137– 155). S. Stevenson and P. Merlo.

Magerman, D. M. (1994). *Natural language parsing as statistical pattern recognition*. Ph.D. thesis, Stanford University.

Marslen-Wilson, W. D. (1973). Linguistic structure and speech shadowing at very short latencies. *Nature*, (pp. 522–523).

Palmer, M., Gildea, D., & Kingsbury, P. (2003). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics, 31*, 71–106.

Staub, A., & Clifton, C. (2006). Syntactic prediction in language comprehension: Evidence from either...or. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 32*, 425–436.

Sturt, P., & Lombardo, V. (2005). Processing coordinate structures: Incrementality and connectedness. *Cognitive Science, 29*, 291–305.

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

Vadas, D., & Curran, J. (2007). Adding noun phrase structure to the penn treebank. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, (pp. 240–247), Prague, Czech Republic. Association for Computational Linguistics.

van Berkum, J. J. A., Brown, C. M., & Hagoort, P. (1999). Early referential context effects in sentence processing: Evidence from event-related brain potentials. *Journal of Memory and Language, 41*, 147–182.

Xia, F., Palmer, M., & Joshi, A. (2000). A uniform method of grammar extraction and its applications. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, (pp. 53–62).