A Broad-Coverage Model of Prediction in Human Sentence Processing

Vera Demberg-Winterfors

Doctor of Philosophy

Institute for Communicating and Collaborative Systems
School of Informatics
University of Edinburgh

2010

Abstract

The aim of this thesis is to design and implement a cognitively plausible theory of sentence processing which incorporates a mechanism for modeling a prediction and verification process in human language understanding, and to evaluate the validity of this model on specific psycholinguistic phenomena as well as on broad-coverage, naturally occurring text.

Modeling prediction is a timely and relevant contribution to the field because recent experimental evidence suggests that humans predict upcoming structure or lexemes during sentence processing. However, none of the current sentence processing theories capture prediction explicitly. This thesis proposes a novel model of incremental sentence processing that offers an explicit prediction and verification mechanism.

In evaluating the proposed model, this thesis also makes a methodological contribution. The design and evaluation of current sentence processing theories are usually based exclusively on experimental results from individual psycholinguistic experiments on specific linguistic structures. However, a theory of language processing in humans should not only work in an experimentally designed environment, but should also have explanatory power for naturally occurring language.

This thesis first shows that the Dundee corpus, an eye-tracking corpus of newspaper text, constitutes a valuable additional resource for testing sentence processing theories. I demonstrate that a benchmark processing effect (the subject/object relative clause asymmetry) can be detected in this data set (Chapter 4). I then evaluate two existing theories of sentence processing, Surprisal and Dependency Locality Theory (DLT), on the full Dundee corpus. This constitutes the first broad-coverage comparison of sentence processing theories on naturalistic text. I find that both theories can explain some of the variance in the eye-movement data, and that they capture different aspects of sentence processing (Chapter 5).

In Chapter 6, I propose a new theory of sentence processing, which explicitly models prediction and verification processes, and aims to unify the complementary aspects of Surprisal and DLT. The proposed theory implements key cognitive concepts such as incrementality, full connectedness, and memory decay. The underlying grammar formalism is a strictly incremental version of Tree-adjoining Grammar (TAG), Psycholinguistically motivated TAG (PLTAG), which is introduced in Chapter 7. I then describe how the Penn Treebank can be converted into PLTAG format and define an incremental, fully connected broad-coverage parsing algorithm with associated probability model for PLTAG. Evaluation of the PLTAG model shows that it achieves the

broad coverage required for testing a psycholinguistic theory on naturalistic data. On the standardized Penn Treebank test set, it approaches the performance of incremental TAG parsers without prediction (Chapter 8).

Chapter 9 evaluates the psycholinguistic aspects of the proposed theory by testing it both on a on a selection of established sentence processing phenomena and on the Dundee eye-tracking corpus. The proposed theory can account for a larger range of psycholinguistic case studies than previous theories, and is a significant positive predictor of reading times on broad-coverage text. I show that it can explain a larger proportion of the variance in reading times than either DLT integration cost or Surprisal.

Acknowledgements

First and foremost I would like to thank Frank Keller for his excellent support and advice during the past years. I very much appreciate his guidance and regular discussions of this work. Frank has always been inspiring, reliable and very accessible. I also want to thank my second advisor Fernanda Ferreira for her valuable advice, and challenging discussions concerning the theoretical assumptions of this work. I'm also grateful to my examination committee, Ted Gibson and Mark Steedman, who made the viva a stimulating experience.

The following people provided valuable feedback to different parts of this work: Manabu Arai, Alexander Koller, Roger Levy, David Reitter, Mark Steedman, Patrick Sturt, Masaya Yoshida and others, in particular all participants and guests at the incrementality reading group.

I would furthermore like to thank Richard Shillcock, Amit Dubey, Martin Pickering and Ewan Klein, who provided helpful comments on my first year proposal and dissertation draft.

This work has also profited from feedback from various reviewers, and from discussions at CUNY 2007 and 2008, AMLaP 2007, 2008, 2009, CogSci 2007 and 2009, TAG+9 2008 and ACL 2010. I have also profited from the attendence of the Graduate Summer School "Probabilistic Models of Cognition: The Mathematics of Mind" at the Institute for Pure and Applied Mathematics (IPAM), UCLA.

I am very grateful to Brian Roark, who kindly made his incremental parser available for this work, and even modified it to enable the computation of prefix probabilities. Special thanks are also due to Asaf Bachrach who provided me with a text collection which was manually annotated with DLT integration cost, thus enabled me to evaluate my implementation of DLT integration cost.

Of course, none of the people mentioned above necessarily agree with all of the claims made in this thesis. All remaining errors and omissions remain my own.

This research was supported by EPSRC grant EP/C546830/1 "Prediction in Human Parsing".

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Vera Demberg-Winterfors*)

To my husband Emanuel and our son Linus.

Table of Contents

Chapter 1

Introduction

This chapter presents the motivation for evaluating sentence processing models on broad-coverage, naturally occurring text and motivates the development of a new model of human sentence processing. It also summarizes the central claims put forward in this thesis and gives an overview of its structure.

1.1 Central Claims

Recent evidence from psycholinguistic experiments suggest that humans predict upcoming structure or lexemes during sentence processing. However, none of the current sentence processing theories model prediction explicitly. The aim of this thesis is to design and implement a cognitively plausible theory of sentence processing which contains a mechanism for modelling syntactic prediction and verification as processes in language understanding.

The thesis puts forward the three claims. The first claim is that evaluation of psycholinguistic theories on broad-coverage data can be a valid additional resource to traditional lab experiments, and that it can provide insights which cannot be obtained from the data acquired in a traditional lab experiment setting.

The second claim is that two previous theories of sentence processing, Surprisal and Dependency Locality Theory explain different aspects of sentence processing.

The third claim in this thesis is that the explicit modelling of prediction and verification is cognitively plausible and provides a framework for combining the different aspects of DLT and Surprisal.

1.2 Motivation

This section motivates the questions addressed and methods used in this thesis.

1.2.1 Evaluation on Naturally Occurring Text

Theories for syntactic processing are usually inspired by observations from very specific structures, such as garden path sentences, relative clauses, verb-final constructions, centre-embedding, ambiguous PP-attachment, idiom processing, case ambiguity, direct object vs. sentence complement ambiguity, etc., and often rather extreme versions of these structures are used to find reliable effects. It is possible that effects observed in carefully controlled lab experiments are rare or absent in naturalistic data such as those found in corpora.

In order for a theory to claim that it is a theory of syntactic processing in humans, it should not only be able to explain the pathologies in human processing, but also processing facilitation and behaviour on a wide variety of structures. Theories should be evaluated on material that humans encounter in their daily lives and not exclusively on unnatural special cases, such as garden paths or difficult constructions that push the boundaries of what humans can process. An important question to ask at this stage is therefore whether the existing theories scale up to naturally occurring, contextualised text, and whether syntactic structures have any measurable influence on such contextualised reading.

Many theories are partial – they were only specified for a subset of what happens in natural language. Applying them to “real” text makes it necessary to complete these theories. Applying sentence processing theories to corpus data helps to assess performance and detect weaknesses, incompleteness and failures of existing theories. Ultimately, theories and computational models of them could be used not only for theoretical insights about sentence processing in humans, but could also be employed in NLP applications.

¿From a corpus, a range of standard eye-tracking measures can be computed just like for experimental materials, but the results hold for naturalistic, contextualised text, rather than for isolated example sentences manually constructed by psycholinguists.

1.2.2 Computational Modelling as a Method

Computational modelling (i.e., implementing theories) is an important methodology in psycholinguistic research. Modelling helps to make many aspects explicit, which otherwise risk to remain underspecified in theories. It allows to observe the effect of assumptions on the global theoretic framework and understand their implications (for example the implications of controversial notions, such as strict incrementality).

Constructing an implementable computational model poses many design questions, which in turn can lead to well-motivated psycholinguistic experiments aiming to answer these questions. Examples from our work are questions addressing the grain size of predictions, whether a main verb should always be predicted, or whether arguments and modifiers should be predicted.

A fully specified model can be used to generate new predictions for unexplored structures which can then be tested in an experimental setting (or on the naturalistic corpus data, of course), and can then provide new insights for refining the model or falsifying assumptions that the model makes.

1.2.3 Why a new Theory of Sentence Processing?

The theory proposed in this thesis differs from existing theories in that it contains an explicit mechanism for prediction and verification. It furthermore assumes that the parsing process is strictly incremental, such that all words in an analysis are always connected under a single node.

Recent experimental evidence for prediction comes from (Kamide et al., 2003; van Berkum et al., 2005; Staub and Clifton, 2006). Modelling prediction and a corresponding verification mechanism can advance our understanding of how they interact with other properties of human sentence processing and how they are reflected in experimental findings. For example, the present work shows that prediction becomes inherently necessary under the assumption of a strictly incremental parsing process.

Finally, current theories of sentence processing can only explain some of the phenomena found in psycholinguistic experiments. Therefore, drawing from those theories and designing a theory that can extend the coverage of previous theories to a wider range of phenomena will constitute an advance in our understanding of human language processing.

1.2.4 Why Focus on Syntax?

The focus of this thesis is on sentence structure. However, processing difficulty is triggered by many other aspects of language understanding, such as lexical access to words, semantic anomalies or discourse. This thesis does not claim that syntax plays a bigger role in processing than the other components. In fact, we expect that low-level effects and artefacts during reading (oculomotor problems, saccade planning, fixation positions of words, shapes and orthographic lengths of words etc.) account for the bulk of the variance in the eye movement record. Current models of reading such as EZ-reader (Reichle et al., 2009) and SWIFT (Engbert et al., 2005) mainly focus on low-level processes, although Cloze probabilities have recently been included into one of the models as a higher-level linguistic predictor to explain regressions during reading (Reichle et al., 2009). However, these models do not allow much introspection or theoretical illumination with respect to understanding how human sentence processing works: Cloze probabilities are a rather coarse measure, and do not disentangle syntactic from lexical or semantic effects. Therefore, it is interesting to investigate whether we can find reading effects of higher-level linguistic processes such as syntax, semantics and discourse using more fine-grained models. Syntax is also well-understood both theoretically and in NLP, and thus provides a good starting point for modelling (rather than semantics or discourse, which usually require some syntactic structure to calculate their representations on).

1.2.5 Relevance

Relevance of this work stems from two aspects. On the one hand, we want to achieve a better understanding of how sentence comprehension works in humans. On the other hand, knowing about what causes processing difficulty for humans can be exploited in human-computer interaction to assess understandability or generate easily understandable text.

The goal of this thesis is not only to build a model that works well in terms of making predictions that correlate with reading time data, but also in particular to build a cognitively plausible model. The model proposed here is cognitively plausible in that it models memory and decay, and draws on concepts from psycholinguistic experiments such as incrementality and prediction.

Possible applications for such a system would be spoken dialogue systems, user adaptation in text generation, e-learning systems or readability checkers. In spoken

dialogue systems, knowing about the processing difficulty a human will incur when hearing or reading a specific sentence could be used to automatically choose among a number of possible ways to formulate the information that is to be conveyed to the user. In addition, deciding on dialogue turn length can be difficult in spoken dialogue systems with elaborated information presentation strategies, (e.g. Demberg and Moore (2006)). A system that knows how difficult a sentence (or paragraph) is could be used to automatically identify the optimal length of a dialogue turn. In a more general context, such a system could be used for choosing the optimal level of difficulty in text generation for a specific group of users (children vs. elderly people vs. adults).

In e-learning (especially foreign language learning), it is even more crucial to choose sentences with an appropriate level of difficulty for the learner, and one way to do this is to measure syntactic difficulty (in addition to choosing an appropriate vocabulary). Furthermore, a "readability checker", similar to a spell-checker, could point out to the author bits in text that contain very complex and difficult to understand syntactic structures.

Another field of applications would emerge if it could be shown that computer programs also have more difficulty processing text that is difficult for humans (one example for this is PP-attachment). Then special strategies could be used to handle bits of text that would be difficult to translate, or we might assign answers from difficult text a lower confidence score in question answering.

1.3 Overview of the Thesis

Chapter 2 provides background information about eye-tracking as a measure of human language processing difficulty. This background is important for understanding the analysis of our main source of data, the Dundee eye-tracking corpus, and the methods used for analysis of this data set (Chapter 3), as well as for the evaluation of models of human processing difficulty (Chapters 4, 5 and 9). Secondly, it positions the theory put forward in this thesis with respect to other sentence processing theories. The detailed description of two theories, dependency locality theory (DLT, Gibson (2000)) and Surprisal (Hale, 2001) provides background for understanding the evaluation of these theories in Chapters 4 and 5, and the design of our theory of explicit prediction, proposed in Chapter 6.

Chapter 3 analyses the properties of the eye-movement information in the Dundee Corpus, a collection of newspaper articles (ca. 50,000 words) for which the eye-movement record of 10 subjects is available. The analysis relates the characteristics of the eye-movement record in the Dundee Corpus to reading characteristics reported in other eye-tracking experiments.

The second part of Chapter 3 explains mixed effects models, and discusses how they can be applied to the Dundee Corpus data.

Chapter 4 represents a proof of concept that broad-coverage data, such as the newspaper texts from the Dundee corpus, can be used as a resource for evaluating theories of sentence processing. We focus on a very specific and reasonably frequent structure, relative clauses. The relative clause asymmetry (object relative clauses are more difficult to process than subject relative clauses) (King and Just, 1991) is a well-established effect. Being able to show that this asymmetry does not only exist in experimental test settings but can also be found in natural language data indicates that broad-coverage texts such as the Dundee corpus can be used as a complementary resource for testing theories, in addition to experimental test suites. It thus motivates our broad-coverage evaluation of sentence processing theories in Chapters 5 and 9.

Chapter 5 evaluates two previous theories of sentence processing, Dependency Locality Theory (DLT) and Surprisal on the broad-coverage data of the Dundee Corpus. To our knowledge, this is the first time that theories of sentence processing have been tested on broad-coverage data. We gain insights about the previous theories' abilities to scale to broad-coverage text and find that in particular, DLT integration cost is not defined on a sufficiently general level to account for general processing difficulty.

Another central finding is the fact that Surprisal and DLT integration cost are uncorrelated, both for arbitrary words in the corpus, and for verbs (for which DLT makes the bulk of its predictions). This result suggests that a complete theory of sentence processing complexity needs to include two mechanisms: a backward-looking one as proposed by DLT, and a forward-looking one as proposed by Surprisal. The analysis thus sets the ground for the development of our own theory (Chapter 6).

Chapter 6 proposes a theory of sentence processing that is designed to be cognitively plausible by implementing fundamental aspects of human sentence processing, such as incrementality and prediction followed by a verification process, and is general enough

for modelling broad-coverage naturally occurring data. The theory also draws from what we learnt from the evaluation of Surprisal and Dependency Locality Theory on the Dundee Corpus. It consists of a parsing process, from which sentence processing difficulty is derived.

The second part of the chapter evaluates five grammar formalisms with respect to their suitability as a basis for implementing the suggested sentence processing theory, and concludes that an incremental version of Tree-Adjoining Grammar (TAG) would satisfy requirements best.

Chapter 7 defines a variant of the Tree-Adjoining Grammar formalism (TAG), called Psycholinguistically Motivated TAG (PLTAG) and relates it to standard TAG. The formalism allows for incremental, fully connected derivation of a sentence, and contains explicit mechanisms for prediction and verification.

Chapter 8 explains the development of a strictly incremental, predictive parser for PLTAG. This chapter first describes the conversion of the Penn Treebank into PLTAG format and the induction of a PLTAG lexicon based on the converted PLTAG tree bank. It then defines an incremental parsing algorithm and a probability model for PLTAG. The parser is evaluated on the Penn Treebank.

The last part of the chapter contains the formalisation of the Linking Theory, which connects the parsing process to processing difficulty predictions generated by the sentence processing theory proposed in Chapter 6.

Chapter 9 evaluates our theory of sentence processing with explicit prediction on data from a range of psycholinguistic case studies and shows that our theory can explain both locality and surprisal effects, which other theories are not able to explain simultaneously. The theory proposed in this thesis is thus more generally applicable than previous theories. Secondly, we evaluate the theory on the broad-coverage, naturally occurring text from the Dundee Corpus and show that our theory is a significant predictor of reading times, and that it can explain a larger amount of the variance than either Surprisal or DLT can. Finally, our sentence processing theory is compared to the alternative sentence processing theories that were outlined in Section 2.2.

Chapter 10 summarizes the main contributions made by this thesis, and gives an outlook on future work.

Chapter 2

Background

This chapter discusses reading times as a measure for processing difficulty in human sentence processing. It outlines the characteristics of reading and factors that influence reading, as well as giving an overview of methods for acquiring reading data, such as eye-tracking. In the second section, previous models of human sentence processing are presented.

These basics about reading, eye-tracking and models of human sentence processing are relevant background for presentation of the Dundee corpus data set and the methodological discussion concerning linear mixed-effects models in Chapter 3, as well as the experiments in Chapters 4, 5 and 9.

2.1 Reading Times as a Measure of Human Sentence Processing

While the high-level goal of my work is to investigate the relationship between syntactic structures and processing difficulty, this thesis deals only with processing difficulty in as far as can be derived from reading times. The principal idea is that reading takes longer at difficult regions in the text, because words are fixated for longer (e.g. it has been found that infrequent words which are arguably more difficult to access lexically, take longer to read than frequent words of the same length), or because parts of a sentence have to be read again. Longer reading times also correlate with e.g. naming latency, another measure that is thought to correlate with difficulty, but which is not specific to reading.

The nature of the relation between language processing and reading is generally

believed to be some more or less strong version of the "eye-mind link" hypothesis which states that people always look at the word they are currently processing. Some people argue against a strong "eye-mind link", saying that reading is a fairly automatic process that proceeds at a very steady speed and is only influenced by low-level visual and oculomotor factors. In this view, syntactic processes only influence the steady reading flow when there is a "total processing breakdown" as in the case of garden paths. In this work, it is assumed that more subtle syntactic processing difficulties can also influence reading times, and can thus be measured in the form of longer reading times and refixations on the words. This work evaluates theories of sentence processing difficulty using linear regressions with reading times as a response variable. This can be interpreted as a strong eye-mind link: syntactic processing difficulty for a word is used as a predictor for reading times at that exact word (however, note that some measures of reading times that aggregate several fixations can possibly partially alleviate the problem). A strict eye-mind link is probably too strong of an assumption, and should be relaxed in future work by integrating the predictions for sentence processing difficulty with a model of eye-movements in reading, see Chapter 10, Section 10.2.6.

A challenge that one faces when using reading times as a correlate of syntactic processing difficulty is that reading times are influenced by a large number of variables. It is difficult to factor out which part of the variation observed in reading times can be attributed to syntactic processing.

Alternative ways of researching processing difficulty are based on grammaticality or plausibility judgements, completion studies (Cloze measure), brain imaging (for example using EEG or fMRI) and visual world studies. An example for grammaticality judgements is the study of *garden path sentences*. A famous example for a garden path sentence is *The horse raced past the barn fell*. At encountering the word *fell*, the most likely analysis of the sentence is incompatible with the continuation *fell*, and a much less likely analysis must be chosen for the prefix (corresponding to the sentence *The horse that was raced past the barn fell*). Garden path sentences illustrate very severe processing difficulty that leads to the total break-down of the analysis process and sometimes can mean that the correct analysis cannot be found, and a grammatical sentence is judged as ungrammatical.

Completion studies are often used in pre-tests to assess whether items have similar probabilities or are similarly easy or difficulty to predict. In a completion study, participants are given a sentence which lacks one or more words and are asked to fill those words in. The Cloze Probability of a word is then the proportion of times that a

particular word was chosen by participants.

During brain imaging studies, language is presented as either text or speech, and participant's neural activity is measured. Examples for well-established effects include the N400 and P600 effects observed in EEG studies. N400 effects have been shown to occur at semantic anomalies (Brown and Hagoort, 1993; van Berkum et al., 1999a) while the P600 effect has been shown to occur at unexpected events, and has often been linked to syntactic problems (Kaan et al., 2000).

A more recent method are visual world studies where people listen to sentences while looking at pictures. The pictures participants fixate can be interpreted as reflecting how participants interpret an ambiguous sentence, or how they expect a sentence to continue.

Limitations of Reading Experiments

Reading can only ever capture a limited amount of information about language processing, since it's a learnt skill and has only been acquired by humans a short time ago (as seen on the evolutionary scale of things). It is different from hearing speech sounds in that a reader can read at his own pace, whereas orally presented words are perceived at a predefined speed. While a reader can go back to passages that he did not understand properly, this is not possible in speech. The speech signal is usually more noisy on the one hand, but also richer because it contains prosodic information that can be used for disambiguation.

2.1.1 Characteristics of Human Reading

When reading, the eye does not move over the text smoothly but in quick jumps. The time while the gaze travels is called a saccade, and usually takes about 25 to 60 milliseconds. During a saccade, no information is taken in: it has been found that people do not even notice light flashes that occur during a saccade. A saccade is usually 7-8 letters far and is assumed to be planned in advance, because fixations very rarely land on punctuation marks, spaces or highly predictable function words, but rather on long words and in the middle of those, where high information density can be expected. The eye also does not move steadily forward in the text, but takes a step backwards roughly every 10 saccades. These backward saccades are also called regressions. The intervals when the eye rests still on a certain point are called fixations. A fixation usually takes 200 to 250 milliseconds.

Figure 2.1 shows an example of a trace of human eye-movements during reading, which were recorded with an eye-tracker. The fixations are marked by blobs on the words, and saccades are shown as fine lines. The small numbers next to the blobs indicate the fixation time in milliseconds. One can observe in this example of reading, how function words like *the*, *or*, *and* received no or fewer fixations than content words.

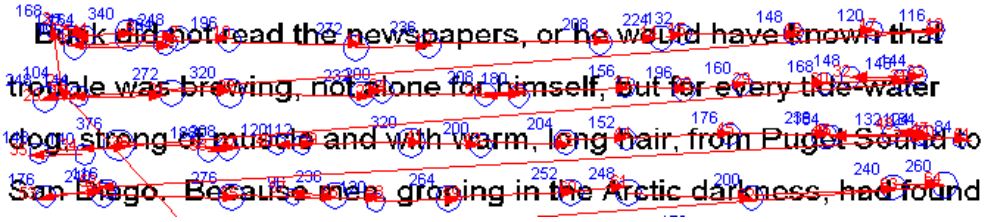


Figure 2.1: Trace of human eye-movements whilst reading recorded with an eye-tracker.

When fixating, information from a window of about 20 characters is taken into account: 3-4 characters to the left and 15 characters to the right. One can also differentiate between the foveal region (about 2° from the fixation point), from which most of the information is extracted and the parafoveal region (about 5° from the fixation point). This data was obtained from studies where part of the text is dynamically obscured, depending on the subject's fixation point, such that the subject can only see a certain part of the text. Obscuring the foveal region has been found to slow down reading considerably, but reading is still possible because enough information can be taken from the parafoveal region (Rayner and Bertera, 1979). Also, it has been shown that obscuring the parafoveal and further away regions slows down reading although subjects are not able to pin down what causes their reading difficulty.

2.1.2 Factors that Influence Reading Times

Saccade lengths, fixation durations and refixation probabilities in reading are influenced by a number of factors, such as letter recognition (fonts, sizes), oculomotor errors (saccade planning errors), word length and frequency effects, fixation landing position and launch distance, syntactic difficulties and effects on the semantic and discourse level, (for a review see Rayner, 1998). When analysing reading times, those of the factors that we are not directly interested in have to be filtered out, in order to obtain the residual variance which cannot be explained by these low-level factors, but potentially by those factors which we are interested in.

The factors that are known to impact reading times are:

- Fonts and sizes

  Fonts are not usually a problem in experimental settings because they will normally be the same for the whole experiment and therefore not represent a potentially confounding factor.

- Beginning or end of sentence or line

  At the end of a line, a long saccade to the beginning of the next line has to be programmed, which may be a more difficult process and hence take more time than programming a short saccade. At the beginning of the line, the saccade target might have been slightly missed, which may lead to a longer fixation, or quick correcting saccade.

- Frequency effects

  Frequency has been found to be an important element in lexical access, with frequent words being read faster than infrequent ones.

- Age of acquisition / familiarity

  Lexical access speeds have also been shown to depend on familiarity with the word (familiar words are read faster), or the age of acquisition: words that were learnt at an earlier age are faster to access than words learnt at a later age. Of course, frequency, familiarity and age of acquisition are strongly correlated, as readers are more familiar with frequent words and have likely acquired them earlier.

- Word length effects

  Length effects are commonly found in reading, with short words being read faster than long words. The length-effects interact with frequency effects (frequent words are usually shorter), age of acquisition (longer or more complicated words are acquired later) or morphological complexity (morphologically complex words are generally longer than simplex words).

- Launch distance

  Vitu et al. (2001); Kennedy et al. (2003) showed that launch distance is also an important contributing factor to fixation duration. The longer the launch distance, the longer the expected fixation.

- Fixation landing position

  The landing position of the fixation within the word has also been shown to have

an important influence on reading times. Typically, words are fixated longest when the saccade lands towards the middle of the word (optimal viewing position). This may seem counter-intuitive as it should be faster to decode a word if viewing acuity is good on most letters of the word, and is hence referred to as the IOVP (inverted optimal viewing position) effect (Vitu et al., 2001).

- Word position within the sentence
  It is sometimes argued that people speed up their pace of reading as the sentence goes on, meaning that words later on in the sentence can be expected to be processed faster than words at the beginning of a sentence.

- Morphological effects (more or less complex words)
  Potentially decomposable words (like *keyhole*) take longer to read than words that are not potentially decomposable but have the same length and frequency.

- Syntactic difficulties
  Words that are syntactically unexpected take longer to read.

- Semantic difficulties
  Semantically mismatching words also take longer to read than words that fit in well semantically.

- Spill-over effects
  If the previous word was difficult to process, longer processing times can also be expected on the current word. This is called a spill-over effect.

- Secondary tasks / concentration / depth of processing
  Finally, interfering tasks or lack of concentration can also influence reading time.

2.1.3 Modelling low-level reading processing

A considerable proportion of the reading times can be attributed to low-level processes for identifying a word and accessing its meaning, which occur before the word is integrated into a larger semantic context.

There are three main approaches among models of eye movements in reading that make different assumptions about the relation between the reading process and the actual fixation position in the text. The one type of models (also known as sequential attention shifts (SAS) models) assumes a fairly close relation between the fixated

words and attention. In this framework, fixation on a particular word means that this word is being processed (with some interference to its directly neighbouring words due to motoric latencies). In such models, the duration and location of a fixation are computed during the reading process and are calculated from cues in the text. The EZ-Reader system is an example for such a model (Reichle et al., 2006). Recently, the EZ-Reader model was also extended to accommodate a component for higher-level linguistic processing, which takes Cloze probabilities as an estimate for syntactic and semantic processing (Reichle et al., 2009).

The second type of model (Guidance by attentional gradients (GAG)) implements a looser relation between attention and fixations, and assumes an underlying mechanism that determines a certain pace and step-width at which the eyes move. An example for this second kind of model is the SWIFT system (Engbert et al., 2005). In SWIFT, the target of a saccade is determined by a stochastic process that is influenced by a word's activation. This activation is dependent on visual and linguistic properties of the words (such as frequency), as well as the eccentricity of the word (how far it is from the current fixation). Fixation duration in SWIFT is modulated by the "inhibition" process which can redirect or inhibit a saccade if lexical access difficulty is encountered.

The third type of model (primary oculomotor control (POC)) mainly models oculomotor processes and doesn't take into account any linguistic processes for modulating the basic low-level visual information (such as word length). The SERIF model (McDonald et al., 2005) is an implemented example of such a non-linguistic approach. One of the main contributions of the SERIF model is the implementation of an anatomic constraint: the foveal split.

All current implemented models of eye-movement in reading focus on oculomotor and lexical access effects, and largely ignore syntactic and semantic processing. A natural way to extend these models would thus be to add more linguistic components. However, such an integration is outside the scope of this thesis. Instead, linear regression models are used to test whether syntactic predictors can account for some of the variance in reading data above and beyond simple low-level oculomotor processes and visual properties of words, which are explained by current models of eye-movements in reading. Two ways of integrating low-level predictions were considered in for this thesis: using the reading time predictions of an existing model as a predictor in the regression model, vs. including the raw values of factors known to influence reading (word length and frequency, launch distance, fixation landing position etc.) as separate predictors. When evaluating the SERIF model predictions on the Dundee Cor-

pus, a regression model that contained all of the raw factors explained a significantly larger amount of the variance in the reading times than the average predictions from the trained SERIF model (to obtain these, the SERIF model was used to simulate the eye-movements of 10 readers on the Dundee Corpus. The simulated reading times were then used as predictors for the real reading times in the regression on the corpus). Given these results, all of the models reported in this work use the single direct influencing factors as predictors instead of predictions from a low-level reading model.

2.1.4 Experimental Methods for the Acquisition of Reading Data

Information about human reading times on test sentences in an experiment can be obtained with different methods. The most common ones are self-paced reading, rapid serial visual presentation and eye-tracking. While rapid serial visual presentation and self-paced reading give the experimenter more control over what exactly the experimental subject is perceiving (e.g. by blocking words together into phrases which are presented at the same time, and not allowing the subject to go back in the sentence), eye-tracking provides a more natural setting for reading.

2.1.4.1 Self-Paced Reading

In self-paced reading (SPR), text is presented in chunks, and the subject has to indicate by pressing a key when they want to go on the next word (or chunk). It has been argued that it is better not to present single words, but larger units, e.g. constituents, because many words (in particular function words) are skipped in natural reading.

There are several ways of conducting self-paced reading studies: words (or constituents) can be presented in the middle of the screen one-by-one, or the whole sentence can be presented with dashes (keeping word length and spaces) and only sequentially revealing a subset of the words at a time.

Major differences between self-paced reading and eye-tracking are that the subject cannot go back to previous parts of the sentence in SPR, but has to keep everything in memory, and don't have parafoveal preview of upcoming words. This might cause reading to be unnatural and lead to artefacts (Bartek et al., 2007), and might cause results from SPR and eye-tracking studies to lead to different results. Finally, SPR has been found to be slower than normal reading, and might thus have different properties (e.g. some processes may occur on the word instead of on the spill-over region), and the additional task of pressing a button might also influence reading behaviour.

2.1.4.2 Eye-tracking

During eye-tracking, the eye-movements are recorded with cameras (often attached to the subject's head) that track the pupils and are calibrated to the screen. The text is presented on a screen, showing one or more lines of text at a time. Eye-tracking has been used in psycholinguistic research since the 70s. Data from eye-tracking studies can help to detect more subtle processing difficulties that are not as obvious as garden paths, and to perform more accurate measurements, e.g. on a word-by-word basis. Processing difficulty manifests itself in longer gaze-durations or re-inspection of parts of the sentence.

The raw data from eye-movement recordings (as shown in Figure 2.1) can be analysed in different ways in order to capture different aspects of processing (e.g. for early processes such as lexical access vs. later processes such as semantic interpretation). Figure 2.2 shows an example of how the eye travelled through the sentence, and explains different eye-tracking measures at the example of the word *himself*. Single fixation measures include first fixation time and second fixation time (not shown in Figure 2.2, but corresponds to only fixation 6). The other fixation measures are multiple fixation measures. Here, "early" measures are often distinguished from "late" measures. An example for an early measure is first pass time (also called gaze duration), which adds up all fixations from first entering a region to first leaving it. Another early measure is the regression path time, which in the example figure would correspond to points 5, 6, 7 and 8. Examples for late measures are second pass time, and total reading time, which is defined as the sum of all fixations on the critical region. A region's skipping rate is the percentage of trials where the first-pass time is zero.

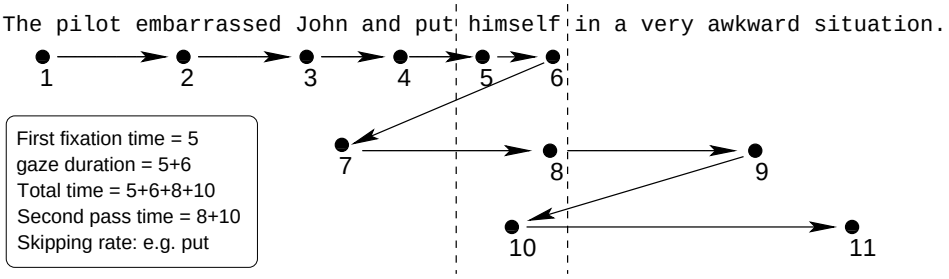


Figure 2.2: Measures for Eye-tracking.

2.1.4.3 Other methods for reading research

Methods used in psycholinguistic research for investigating lexical access and word identification processes are for example *rapid serial visual presentation* where words are presented at a high rate of up to 1000 words per minute. The disadvantage of this method is that it is also unnatural and has problems similar to self-paced reading. Another method is to make the subject *read aloud* while recording eye-movement to measure the distance between fixation and pronounced word and record reading errors and recovery strategies.

Further tasks that primarily test for lexical access are the *naming task*, where the time between presentation of a word and the onset of the pronunciation of the word is measured, the *lexical decision task*, where participants have to tell whether a string is a valid word or not, and the *semantic categorisation task* that assesses how long it takes the participant to retrieve the semantics of a word.

2.2 Models of Human Sentence Processing

Low-level reading processes cannot explain all of the variance in the eye-movement data, and it has been shown that some of the reading time effects are due to high level syntactic or semantic processing difficulties. This thesis focuses on explaining the variance in the reading time data which cannot be explained by the low-level processes.

A number of theories have been proposed to account for processing difficulty effects that are due to syntax and semantics. Ambiguity has often been looked at as a primary source of syntactic processing difficulty, especially in garden path sentences, where ambiguity causes the parser to re-analyse a sentence and thus leads to processing delays. However, ambiguous structures have also been shown to sometimes facilitate reading (van Gompel et al., 2005), at least if the ambiguity remains unresolved. On the other hand, increased processing difficulty has also been observed in completely unambiguous structures. For these cases, the cause of processing difficulty is usually attributed to complexity, or unexpected syntactic or semantic events.

Sentence processing theories can roughly be categorised into theories explaining processing difficulty through ambiguity, and ones explaining difficulty based on complexity. Ambiguity has been argued to lead to difficulty either through competition arising between alternative analyses (competition-based models), through unexpected events, i.e. when the previously most likely analysis becomes improbable or impos-

sible given evidence from new input (reanalysis theories, frequency-based theories, Surprisal, transitional probabilities) or when ambiguity is reduced due to a word providing a lot of information as to how the sentence will continue (Entropy). Processing difficulty has been linked to sentence complexity through a large number of open dependencies that need to be remembered and later retrieved (Dependency Locality Theory, Memory and Activation).

2.2.1 Early approaches

The earliest approaches to explaining processing difficulty primarily focused on difficulty caused by ambiguity. Such early models defined a set of cognitively motivated constraints to decide which structures should be preferred over others.

The earliest approach to modelling human parsing was via serial models. In garden path sentences, such a model would explain delays through extensive backtracking and trying to match parses to the input.

Marcus (1980) assumed that human parsing was serial and suggested fixed constraints, such as a restricted length context window to predict the existence of garden path sentences. According to this theory, easy re-analysis is only possible if the change in interpretation affects the last x words only. Ambiguities that are due to an earlier word lead to garden paths sentences. However, many argued against this theory by showing that there are both garden path sentences that generate ambiguities within such a window and sentences where the ambiguous point is further away but that do not lead to garden paths (see for example (Jurafsky, 1996)).

As an alternative to serial models, parallel parsing models were developed based on experimental evidence suggesting that lexical items and idioms are accessed in parallel (Swinney and Cutler, 1979). In parallel models, multiple interpretations (both structural and lexical) are maintained simultaneously. The first constraint-based parallel parsing models were non-probabilistic and assumed the existence of a number of constraints. The function of these constraints is to rank the alternative parses for a sentence, such that some structures would be predicted to be preferred over others. Very strongly dis-preferred structures are pruned in this model, in order to provide a mechanism to account for garden path sentences. The most important constraints used in serial non-probabilistic models are based on locality preferences (e.g. Right Association (Kimball, 1973), Local Association (Frazier and Fodor, 1978), Late Closure (Frazier, 1978), Final Arguments (Ford et al., 1982), the Graded Distance Effect

(Schubert, 1984), Attach Low and Parallel (Hobbs and Bear, 1990) and the Recency Preference (Gibson, 1991)).

2.2.2 Dependency Locality Theory

Dependency Locality Theory (DLT), suggested by Gibson (1998), explains processing difficulty independent of ambiguity. Instead, processing difficulty in DLT is caused by the cost of the computational resources consumed by the processor. Two distinct cost components can be distinguished: (i) *integration cost* associated with integrating new input into the structures already built at a given stage in the computation, and (ii) *memory cost* involved in the storage of parts of the input that may be used in parsing later parts of an input. In our implementation (see Chapters 4 and 5), we will focus on integration cost, as "reasonable first approximations of comprehension times can be obtained from the integrations costs alone, as long as the linguistic memory storage used is not excessive at these integration points" (Gibson, 1998, p. 19f). This is a safe assumption for our studies, as we use corpora of carefully edited newspaper text, which are unlikely to incur excessive storage costs (in contrast to artificially constructed experimental materials). Gibson defines integration cost as follows:

(1) **Linguistic Integration Cost**

The integration cost associated with integrating a new input head h_2 with a head h_1 that is part of the current structure for the input consists of two parts: (1) a cost dependent on the complexity of the integration (e.g. constructing a new discourse referent); plus (2) a distance-based cost: a monotone increasing function I(n) energy units (EUs) of the number of new discourse referents that have been processed since h_1 was last highly activated. For simplicity, it is assumed that I(n) = n EUs. (Gibson, 1998, p.12f)

According to this definition, integration cost is dependent on two factors. First, the type of element to be integrated matters: new discourse referents (e.g., indefinite NPs) are assumed to involve a higher integration cost than old/established discourse referents, identified by pronouns. Second, integration cost is sensitive to the distance between the head being integrated and the head it attaches to, where distance is calculated in terms of intervening discourse referents.

As an example, consider the subject vs. object relative clause example in Figure 2.3. At the embedded verb *attacked* in the subject relative clause, two integrations take

place: the gap generated by the relative pronoun *who* needs to be integrated with the verb. The cost for this is I(0), as zero new discourse referents have been processed since the gap was encountered. In addition, the embedded verb *attacked* needs to be integrated with its preceding subject. Again, this is a free integration since no discourse referent occurs between the verb and the subject NP. However, there is a cost for building a new discourse referent (the embedded verb itself[1]), leading to a cost of I(1). The total cost at *attacked* is therefore I(1). This is illustrated in Figure 2.3, which depicts the dependencies that are built, and the integration costs per word that are incurred.

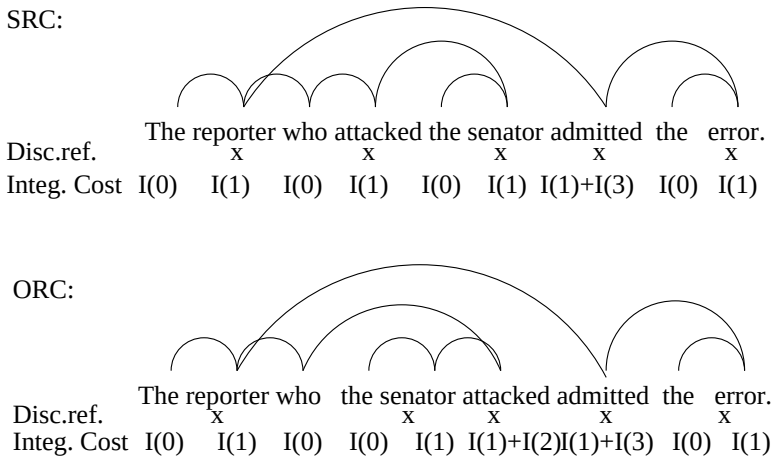


Figure 2.3: An example of integration cost computations: subject relative clauses (SRC) vs. object relative clauses (ORC), with word-by-word mark-up for discourse referent and integration costs. The links between the words represent syntactic dependencies.

At the verb *attacked* in the object relative clause, three structural integrations take place: (1) integration with the subject NP *the senator*: no integration costs occur since no new discourse referents occurs inbetween the verb and the NP, (2) an empty category for the relative pronoun is integrated, but again, the integration is local and no costs occur, (3) the object position empty category is co-indexed with the preceding relative pronoun *who*. There is an integration cost of I(2) for this step due to the two discourse referents, *attacked* and *the senator* which occurs in between. In addition, there is a cost of I(1) for constructing the discourse referent at *attacked*, which leads to a total integration cost of I(1) + I(2) at the embedded word of the object relative clause. So overall, DLT predicts that the verb of an object relative clause is more difficult to process than that of a subject relative clause. Note that Gibson assumes that the inte-

[1] DLT assumes that verbs introduce event discourse referents.

gration cost function is identity, i.e., I(n) = n. However, other functions are possible here; we will return to this issue in our implementation of integration cost in Chapter 5, Section 5.3.2.

For assessing the second component, memory cost (also referred to as storage cost), it is necessary to determine the subcategorizations of a word in order to count how many open dependencies need to be maintained at each point in time. There is a cost for storing in memory each open dependency, which is modulated by how many discourse referents were introduced since the occurrence of the dependent.

Integration costs and memory cost interact through the concept of *energy units*, *memory units* and *time units*. There is only a limited number of energy units available at each point in time, so working memory resources can be used up by having to remember many dependencies (thus using up lots of memory units), in which case there will be less resources for actual integrations (as measured using integration cost), in turn causing them to take more time. The relationship between energy units, memory units and time units was formalised as $EU = MU * TU$. In the case of ambiguity, analyses that require fewer energy units are preferred.

DLT has been shown to account for a range of linguistic effects including the SRC/ORC processing difficulty asymmetry, difficulty of centre embeddings, cases of processing breakdown, filler-gap dependencies, heavy NP shift and extraposition.

2.2.3 Frequency-based Theories

Frequencies have been found to be an important correlator for processing difficulty and reading time. Reading times are in general longer on infrequent words than they are on frequently occurring words. Frequency effects do arguably not only occur for lexical access but also for syntactic processing: If a sentence is ambiguous, humans have been found to process the more frequent analysis faster and to display processing difficulty if the infrequent analysis turns out to be correct.

Jurafsky (1996) first proposed to use probabilistic context free grammars (PCFGs) to estimate probabilities of alternative analyses and used the probabilities to explain garden path sentences: Only the most probable parses (according to the PCFG) would be kept in memory. The improbable ones were pruned using beam search, which discards highly improbable analyses. For interpretations that were pruned, the parser would have to backtrack, which explains the processing difficulty for garden path sentences.

Crocker and Brants (2000) argue that PCFGs can be used to do broad-coverage parsing better than models that only take into account single manually defined constraints, as the early approaches did.

A question that arises in the construction of probability-based models is how to combine probabilities from different sources, such as probabilities from parsing, n-grams and valency. Narayanan and Jurafsky (2002) use a belief network with probabilities from a range of different sources. A belief network is more powerful than a PCFG but it is harder to justify the contributing factors and their relationship to one-another.

2.2.4 Surprisal

An alternative measure of syntactic complexity has been proposed by Hale (2001) in the form of Surprisal. Surprisal assumes a parallel parser, which builds structures incrementally, i.e., it constructs all possible syntactic analyses compatible with the input string on a word-by-word basis.[2] Intuitively, Surprisal measures the change in probability mass as structural predictions are proven wrong when a new word is processed. If the new word invalidates predictions with a large probability mass (high Surprisal), then high processing complexity is predicted, corresponding to increased reading time. If the new word only invalidates predictions with a small probability mass (low Surprisal), then we expect low processing complexity and reduced reading time.

Technically, Surprisal can be defined using the conditional probability $P(T|w_1 \cdots w_k)$, i.e., the probability of a tree T given the sentence prefix $w_1 \cdots w_k$. This is the probability that T is the correct tree, given that the string of word w_1 to word w_k has been encountered. Surprisal is then defined as the change in the conditional probability distribution from w_k to w_{k+1}. As Levy (2008) shows, this can be formalised using the Kullback-Leibler divergence (relative entropy). The Kullback-Leibler divergence between two probability distributions P and Q is defined as:

$$D(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{2.1}$$

The Surprisal at encountering word w_{k+1} then corresponds to the Kullback-Leibler divergence between $P(T|w_1 \cdots w_{k+1})$, i.e., the probability distribution of all syntactic

[2] While Surprisal is compatible with a fully parallel parser, it does not necessarily require one. It is possible to compute the probabilities of a limited set of analyses and then use these to track changes in the probability distribution. In fact, the (Roark, 2001a) parser used in this paper performs beam-search, i.e., does not compute all possible analyses, and thus we reply on such a limited-parallelism version of Surprisal.

trees that are consistent with words $w_1 \cdots w_{k+1}$, and $P(T|w_1 \cdots w_k)$, the probability distribution of the trees that are compatible with the prefix $w_1 \cdots w_k$:

$$S_{k+1} = \sum_T P(T|w_1 \cdots w_{k+1}) \log \frac{P(T|w_1 \cdots w_{k+1})}{P(T|w_1 \cdots w_k)} \tag{2.2}$$

This expression can be simplified using the following fact:

$$P(T|w_1 \cdots w_k) = \frac{P(T, w_1 \cdots w_k)}{P(w_1 \cdots w_k)} = \frac{P(T)}{P(w_1 \cdots w_k)} \tag{2.3}$$

This equation holds because we know that each tree in T contains the words $w_1 \cdots w_k$, therefore $P(T, w_1 \cdots w_k) = P(T)$. We can now substitute Equation (2.3) into Equation (2.2). We can then simplify the definition of Surprisal using the fact $\sum_T \frac{P(T)}{P(w_1 \cdots w_{k+1})} = 1$ (the probabilities of all syntactic trees given a particular prefix sum up to 1), and performing some straightforward logarithmic transformations:

$$\begin{aligned} S_{k+1} &= \sum_T \frac{P(T)}{P(w_1 \cdots w_{k+1})} \cdot \log \frac{\frac{P(T)}{P(w_1 \cdots w_{k+1})}}{\frac{P(T)}{P(w_1 \cdots w_k)}} = 1 \cdot \log \frac{P(w_1 \cdots w_k)}{P(w_1 \cdots w_{k+1})} \\ &= -\log \frac{P(w_1 \cdots w_{k+1})}{P(w_1 \cdots w_k)} = -\log P(w_{k+1}|w_1 \cdots w_k) \end{aligned} \tag{2.4}$$

This derivation shows that the Surprisal S_{k+1} at word w_{k+1} corresponds to the negative logarithm of the conditional probability of w_{k+1} given the sentential context $w_1 \cdots w_k$. This is an important simplification, as it means that Surprisal can be computed without making representational assumptions (i.e., the syntactic tree T does not figure in the definition of Surprisal). In practice this means that a number of ways of computing Surprisal are possible, utilising either simple probabilistic models of language (such as *n*-gram models) or more sophisticated ones, such as probabilistic context-free grammars (PCFGs).

Surprisal can be reformulated in terms of the *prefix probabilities* of words w_k and w_{k+1}, which can be obtained easily from a PCFG. The prefix probability of a word w_k is obtained by summing the probabilities of all trees T that span from w_1 to w_k:

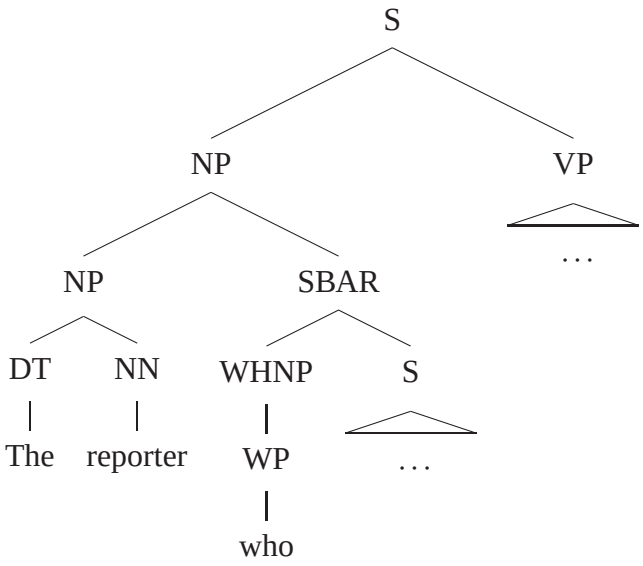
$$P(w_1 \cdots w_k) = \sum_T P(T, w_1 \cdots w_k) \tag{2.5}$$

The formulation in Equation (2.4) is therefore equivalent to a formulation that uses prefix probabilities:

$$S_{k+1} = -\log \frac{P(w_1 \cdots w_{k+1})}{P(w_1 \cdots w_k)} = \log \sum_T P(T, w_1 \cdots w_k) - \log \sum_T P(T, w_1 \cdots w_{k+1}) \tag{2.6}$$

Surprisal S_{k+1} at word w_{k+1} thus corresponds to the difference between the logarithm of the prefix probabilities of word w_k and w_{k+1}. We give an example that illustrates how prefix probabilities can be computed using a PCFG. In a PCFG, each context-free grammar rule is annotated with its probability, as in Figure 2.4. The rule probabilities are then used to calculate the prefix probability of a word.

For example, if w_{k+1} is the word *who* in the example in Figure 2.4, then the prefix probability $\sum_T P(T, w_1 \cdots w_{k+1})$ is the sum over the probabilities of all possible trees that include the prefix $w_1 \cdots w_{k+1}$, where each tree probability is computed as the product of all the rules that are needed to build the tree (Figure 2.4 shows only one such tree).



Example	**Rule**	**Rule probability**
The reporter who ...	S → VP NP	$p = 0.6$
The reporter who ...	NP → NP SBAR	$p = 0.004$
The reporter	NP → DT NN	$p = 0.5$
The	DT → the	$p = 0.7$
reporter	NN → reporter	$p = 0.0002$
who ...	SBAR → WHNP S	$p = 0.12$
who	WHNP → WP	$p = 0.2$
who	WP → who	$p = 0.8$

Figure 2.4: Example derivation of prefix *The reporter who* and rules from a probabilistic context free grammar (PCFG) that would be needed in order to calculate its prefix probability.

Levy (2008) evaluated Surprisal on a range of syntactic processing difficulty phenomena and found that it can correctly account for anti-locality effects in German, facilitating ambiguity and subject preference in German, but that it cannot account for locality effects found in English relative clauses, digging-in effects or local coherence effects (see Section 9.1 for a discussion and overview of these psycholinguistic effects).

2.2.5 Transitional Probabilities

Recently, it has also been shown that information about the sequential context of a word can influence reading times. In particular, McDonald and Shillcock (2003b) present data extracted from an eye-tracking corpus (a smaller corpus than the Dundee corpus used here) that show that forward and backward transitional probabilities are predictive of first fixation and gaze durations: the higher the transitional probability, the shorter the fixation time.

By *forward transitional probability* McDonald and Shillcock (2003b) refer to the conditional probability of a word given the previous word $P(w_k|w_{k-1})$. This captures the predictability of the current word given a one-word context. For example, the probability of the word *in* given that the previous word was *interested* is higher than the probability of *in* if the last word was *dog*. The *backward transitional probability* is the conditional probability of a word given the next word $P(w_k|w_{k+1})$. This provides an estimate of how predictable the current word is given the next word, e.g., of how probable it is to see *interested* or *dog* currently, given the next word is *in*. A possible interpretation of why material that is further away in the text can benefit the current word and lead to shorter reading times for words with high backward transitional probabilities are preview effects and backward saccades. These corpus results are backed up by results demonstrating the role of forward transitional probabilities in controlled reading experiments (McDonald and Shillcock (2003a); but see Frisson et al. (2005), who equate transitional probability and Cloze predictability and do not find any effects of transitional probability).

It is interesting to note that the forward transitional probability $P(w_k|w_{k-1})$ is a simple form of Surprisal, viz., one that takes into account only the previous word w_{k-1}, rather than the whole prefix $w_1 \cdots w_{k-1}$ (see Equation (2.4)). Another difference is that forward transitional probabilities are estimated using word bigrams, while Surprisal is typically estimated using syntactically generated probabilities, based on

Equations (2.5) and (2.6). We will return to this issue in the context of our discussion of Surprisal in the Dundee Corpus in our chapter about broad-coverage evaluation of Surprisal in Section 5.5.

2.2.6 Entropy

Another suggestion for measuring difficulty in processing sentences based on the changes in probability distributions of analyses when processing a sentence is *Entropy* (Hale, 2003, 2006) which quantifies the uncertainty about the rest of the sentence.

The entropy of the probability distribution over the set of all possible sentences S with length n is defined as

$$H = -\sum_{w_1^n \in S} P(w_1^n) \log(P(w_1^n)).$$

When words are processed, these distributions change (as there are many sentences in the set of all possible sentences that are not compatible with the seen input). The entropy at a word i is therefore

$$H(i) = -\sum_{w_1^n \in S} P(w_{i+1}^n | w_1^i) \log(P(w_{i+1}^n | w_1^i)).$$

The reduction in entropy through processing the next word is then

$$\Delta H(i+1) = H(i) - H(i+1).$$

Positive ΔH correspond to a decrease in entropy, hence meaning that the current word has diminished our uncertainty about how the sentence is going to continue. Non-negative ΔH are used to predict reading times at each word.

Hale (2006) showed that entropy can explain linguistic phenomena such as the accessibility hierarchy. Recently, entropy as a measure of processing difficulty has been evaluated as a broad coverage model (Roark et al., 2009; Frank, 2010), showing that it can be a significant positive predictor of reading times.

2.2.7 Competition-Based Models

Competition-based models focus on processing difficulty caused by ambiguity. The main idea in competition models (McRae et al., 1998) is that alternatives compete against one another (in terms of frequency, structure etc.) until one of the alternatives reaches criterion. The system then settles on one analysis. If this analysis turns

out to be incorrect, the system has to "change its mind" later on; i.e. switch to a competing analysis. This switching to an alternative analysis manifests itself as a garden path effect. The competition process also takes up time and processing resources: people are assumed to read more slowly if there are two closely concurrent alternatives than when one is very plausible and one very implausible (this is quite similar to an entropy-based approach).

An interesting finding in (van Gompel et al., 2005; Traxler et al., 1998) is the fact that there can be a processing advantage if a sentence is ambiguous with respect to e.g. PP-attachment, so that the ambiguous sentence can be processed faster than the unambiguous sentence. This finding does not fit with some of the common assumptions in human sentence processing, such as that processing difficulty would be caused by resolving ambiguities. In particular, finding that the ambiguous case is less difficult to process seems to provide evidence against competition-based models.

However, Green and Mitchell (2006) argued that facilitation on ambiguous structures can be explained by an averaging effect where difficulty does not occur due to competition but only due to backtracking (i.e. when the system has to change its mind and switch to an alternative analysis). Across different participants, different initial interpretations were adopted and always cause some of the people to re-analyse, whereas in ambiguous structures, everybody can keep their initial analysis and average reading times are therefore shorter.

Competition-based models can be divided into short-lasting competition models and long-lasting competition models (van Gompel et al., 2005). Long-lasting competition models claim that competing syntactic analyses are kept in parallel throughout an ambiguous region until some disambiguating element is encountered. Short-lasting competition models assume that there are initially alternative analyses which are activated in parallel, but one of them rapidly wins and receives much more activation than its alternative, which causes this one analysis to be adopted sometimes even before the disambiguating region is reached.

Tabor et al.'s (1997) *Visitation Set Gravitation* (VSG) model makes use of dynamical systems theory, and also derives processing difficulty from competition between analyses. It is implemented as a simple recurrent network and a "gravitation module" which clusters similar states in the network. These clustered states correspond to the different competing analyses of a sentence. Processing difficulty is then predicted to be proportional to the time the gravitation module needs to gravitate on one cluster, i.e. decide on a particular analysis. Furthermore, difficulty occurs when new evidence

means that a different cluster becomes more prominent (Tabor and Tanenhaus, 2001). VSG can explain thematic expectations, competition effects and the main clause bias in contrast with reduced relative clauses.

2.2.8 Memory and Activation

Lewis and Vasishth (2005) proposed a model of sentence processing that uses the cognitive architecture ACT-R. This is attractive, in that ACT-R implements cognitively plausible mechanisms, in particular working memory architecture, and has also been used to model a large range of other cognitive processes.

The memory and activation model explains many of the established processing phenomena through memory retrieval effects. The underlying mechanisms of memory retrieval are rehearsal, spreading activation and decay. Their implementation uses left-corner parsing to determine top-down predictions about what types of words or structures are needed to build a sentence, simultaneously with bottom-up evidence for what words are encountered in the input. When a word is retrieved from memory, its activation is boosted (this explains e.g. lexical frequency effects: items that are retrieved very often have higher activation), at the same time there is a steady activation decay according to the power law of forgetting which is applied to all of the items in memory.

The model accounts for locality effects (like the English SRC/ORC asymmetry and centre embedding) through decay and resulting lower activation of words that need to be retrieved for integration after seeing a lot of intervening material. It can also account for some anti-locality effects through activation of the head through intervening arguments. Furthermore, the theory can explain interference effect (retrieval is hindered by activation of similar items) and storage load effects (if more items need to be stored, there are also more interference effects at retrieval).

2.3 Summary and Conclusions

The first part of this chapter has provided background for measures of sentence processing difficulty, discussing in particular reading times as a correlate for processing difficulty. In a comparison of alternative methods of gathering reading time data, this chapter has argued that eye-tracking is the most naturalistic method among them. Eye-tracking measures are used for a range of evaluations in Chapters 4, 5 and 9.

The second part of this chapter discussed existing theories of sentence processing. Out of these theories, Surprisal and DLT integration cost are most relevant for this thesis (in particular for Chapters 4 and 5) and are therefore explained in detail. Surprisal and DLT belong to different categories of sentence processing theories: Surprisal explains processing difficulty through unexpected events, while DLT predicts processing difficulty when many dependencies need to be stored in memory simultaneously, and when long distance dependents have to be retrieved from memory for integration.

Chapter 3

Data and Analysis Methods

The first section of this chapter describes the properties of a large eye-tracking corpus, the Dundee Corpus, which is the data resource used for the regression analyses described in Chapters 4, 5 and 9. We show that all of the standard reading effects can be found in the Dundee Corpus data and point out some ways in which the naturally occurring text differs from experimental items. The studies reported in this thesis represent the first time that such a collection of naturally-occurring text has been used to evaluate models of higher-level linguistic processes such as syntactic processing difficulty.

The second part of the chapter discusses linear mixed-effects regression models, which are going to be used as a method of evaluating sentence processing models, in Chapters 4, 5 and 9. All of the experiments and statistical analyses presented here and in the following chapters were computed using R, an open source programming language and environment for statistical computing (R Development Core Team, 2007).

3.1 The Dundee Corpus

The Dundee Corpus (Kennedy and Pynte, 2005) contains English and French newspaper articles, which are each annotated with the eye-movement data. This section focuses on the properties of the English portion of the Dundee Corpus (the French subcorpus was not used in this work). The English corpus contains 20 approximately equally long articles from *The Independent* newspaper. In total, it consists of 51,502 tokens[1] and 9,776 types. The texts were split into 40 five-line screens for presenta-

[1]The token number refers to tokens as tokenized in the Dundee Corpus for presentation to the participants, i.e., punctuation marks are attached to the words. If words and punctuation marks are counted

tion to the readers during eye-tracking. It is annotated with the eye-movement records of 10 English native speakers, who each read the whole corpus, and answered a set of comprehension questions after each text. These eye-tracking data were acquired using a Dr. Boise eye-tracker, which recorded the movements of the right eye with a sampling rate of 1 ms and a spatial accuracy of 0.25 characters.

Before carrying out our analyses, we excluded all cases in which the word was the first or last one of the line, and also all cases where the word was followed by a any kind of punctuation. This eliminates any wrap-up effects that might occur at line breaks or at the end of sentences. Furthermore, we excluded all words that were in a region of four or more adjacent words that had not been fixated, since such regions were either not read by the participant or subject to data loss due to tracking errors. This left us with 385,467 words.

In the first part of this section, distributions of the reading measures in the Dundee Corpus are shown and problems with the data, as well as particularities of the corpus are discussed. The second subsection looks at the distributions of non-syntactic oculomotor and lexical explanatory variables in the corpus, and shows the typical reading effects like the IOVP effect[2], the length and the frequency effect. Finally, the third subsection shows the distribution of a number of syntactic explanatory variables that were tested in this thesis. More sophisticated analyses and regression models are described in later chapters.

3.1.1 Distributions of Reading Measures

The reading measures described in this section are first fixation duration, first pass duration and total reading time. We focus on these measures here because these eye-tracking measures are also reported during later experiments, as they seemed most informative – first fixation duration and first pass duration are early measures and often assumed to show lower-level effects and fast higher-level effects, while total reading time is a later measure and is thought to reflect higher-level linguistic processing, which we are primarily interested in in this work. Furthermore, skipping, refixation probabilities and regression probabilities are discussed. Finally, the problems of track loss, which is quite common in the Dundee corpus, and inter-subject variability are addressed. Subject variability is particularly important here, because the Dundee corpus was only read by 10 subjects, which is a rather small number when compared to exper-

separately, then there are a bit more than 56k words in the corpus.

[2]For a definition of the effect please see Section 3.1.2.2.

imental settings or other corpora that were used for investigating reading behaviour, such as the Potsdam Sentence Corpus (Kliegl et al., 2006), which was read by more than 250 subjects.

3.1.1.1 First Fixation Duration, First Pass Duration and Total Reading Time

We start by inspecting the distribution of first fixation, first pass and total reading times in the Dundee Corpus. Figure 3.1 shows three histograms for first fixation durations, Figure 3.2 shows the equivalent histograms for first pass durations and Figure 3.3 for total reading times.

Firstly, we can see in the top subfigures of Figures 3.1 to 3.3 that many of the data points have a reading time value of 0. These words were skipped during reading. If we included these values into our regressions, they would cause non-normality and heavily influence the regression estimations. The problem with these cases stems from the fact that there is no smooth distribution of shorter and shorter reading times until they equal zero. Furthermore, it may be questionable, whether the meaning of skipping a word would be the same as fixating it for an incredibly short time. Therefore, all the regression models in this thesis are only run on fixated words; skipping can be dealt with in separate, logistic regression models.

Inspecting the middle subfigure of Figures 3.1 to 3.3, it becomes clear that the data is not exactly normally distributed: the plotted normal distribution does not fit the data very well (with the most severe mismatch for total reading times). The empirical data is skew, with a long tail to the right because fixation durations or reaction time can become very long, but never shorter than zero. Furthermore, there is a sudden cut-off at the left tail at about 60ms, which is due to internal settings of the eye-tracker. Fixation shorter than 60ms are regarded as microsaccades or measurement errors and therefore aggregated with the previous or following saccade. This non-normality of reading times however comes it no surprise: It is already well-known from the literature that reading times (and all other kinds of reaction times) are usually skew to the right (Ratcliff, 1979, p. 447). Non-normality can potentially cause problems in regression models that assume normal distribution of the response variable.

One way of dealing with non-normal distributions is to transform the data, for example by using log-transformed reading times. The log-transformed first fixation durations, first pass durations and total reading times are fitted much better by the normal distributions, see bottom plots in Figures 3.1 to 3.3. The other solution is to change assumptions and not use the normal distribution (but e.g. the gamma distribution) in

regressions. Centre plots in Figures 3.1 to 3.3 show that the gamma distribution (indicated by the dashed red line) fits the data much better than the normal distribution. Note that the normal and gamma distribution almost completely overlap for the log transformed reading times, suggesting that it would be unnecessary to use the gamma distribution if reading times are log transformed. We can also make the same argument more formally: In the literature (Coolican, 2004, p. 292), a common rule of thumb for deciding whether a distribution seriously differs from the normal distribution is by checking whether the skew of a distribution is significantly different from 0. The skew of a distribution is calculated as $skew = \frac{\frac{\sum y-\bar{y}}{n}}{(\sqrt{s^2})^3}$ where n is the number of data points and $\sqrt{s^2}$ is the standard deviation. The skew is then tested for significance of being different from 0 by dividing it by its standard error. The skew of first fixation times is 1.37, which corresponds to t-value 3.17 and is significantly different from zero. On the other hand, the skew of log-transformed first fixation times is -0.66, with t-value -1.5 and is thus not significantly different from 0. This indicates that log transformed reading times would not violate the assumption of a normally distributed response variable in a regression model. We will return to this issue in more detail in Section 3.2.

¿From the plots in Figures 3.1 to 3.3, it is also evident that tails become heavier and heavier from first pass to total reading time. This is due to the aggregation of multiple fixations. As we will discuss in Section 3.2, the variance in the reading time data therefore also increases with larger means. This phenomenon is known as heteroscedasticity.

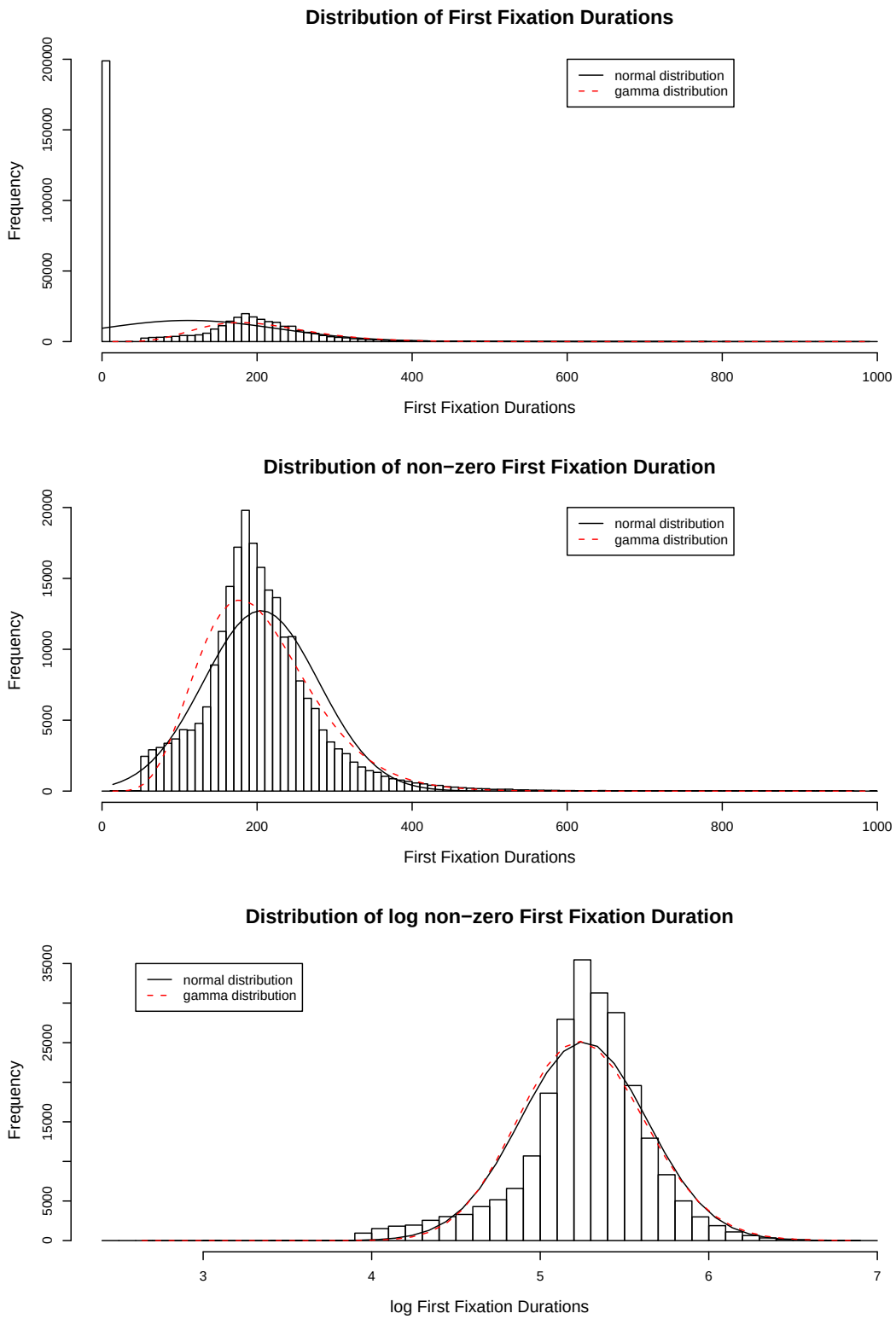


Figure 3.1: Distribution of First Fixation Durations

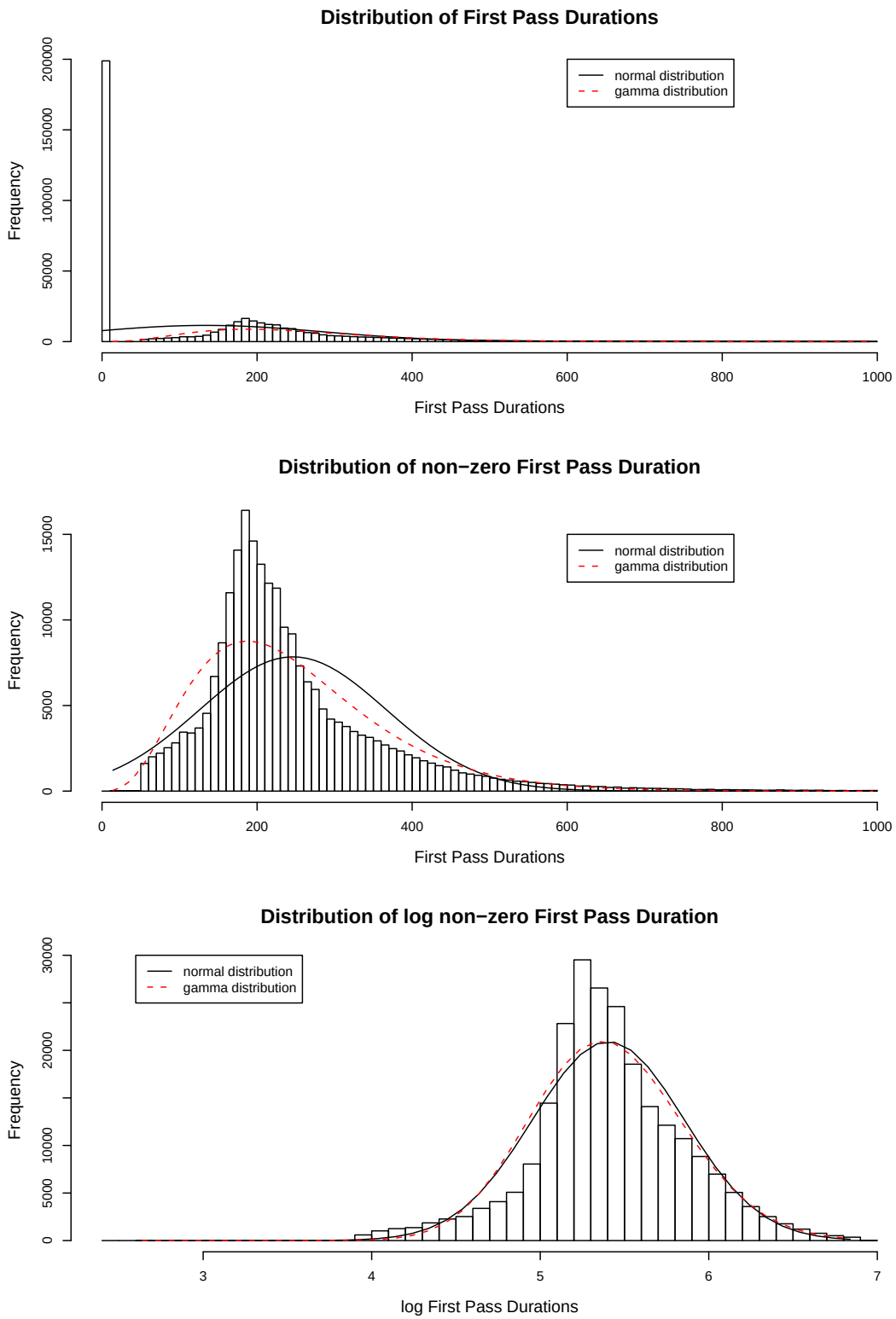


Figure 3.2: Distribution of First Pass Durations

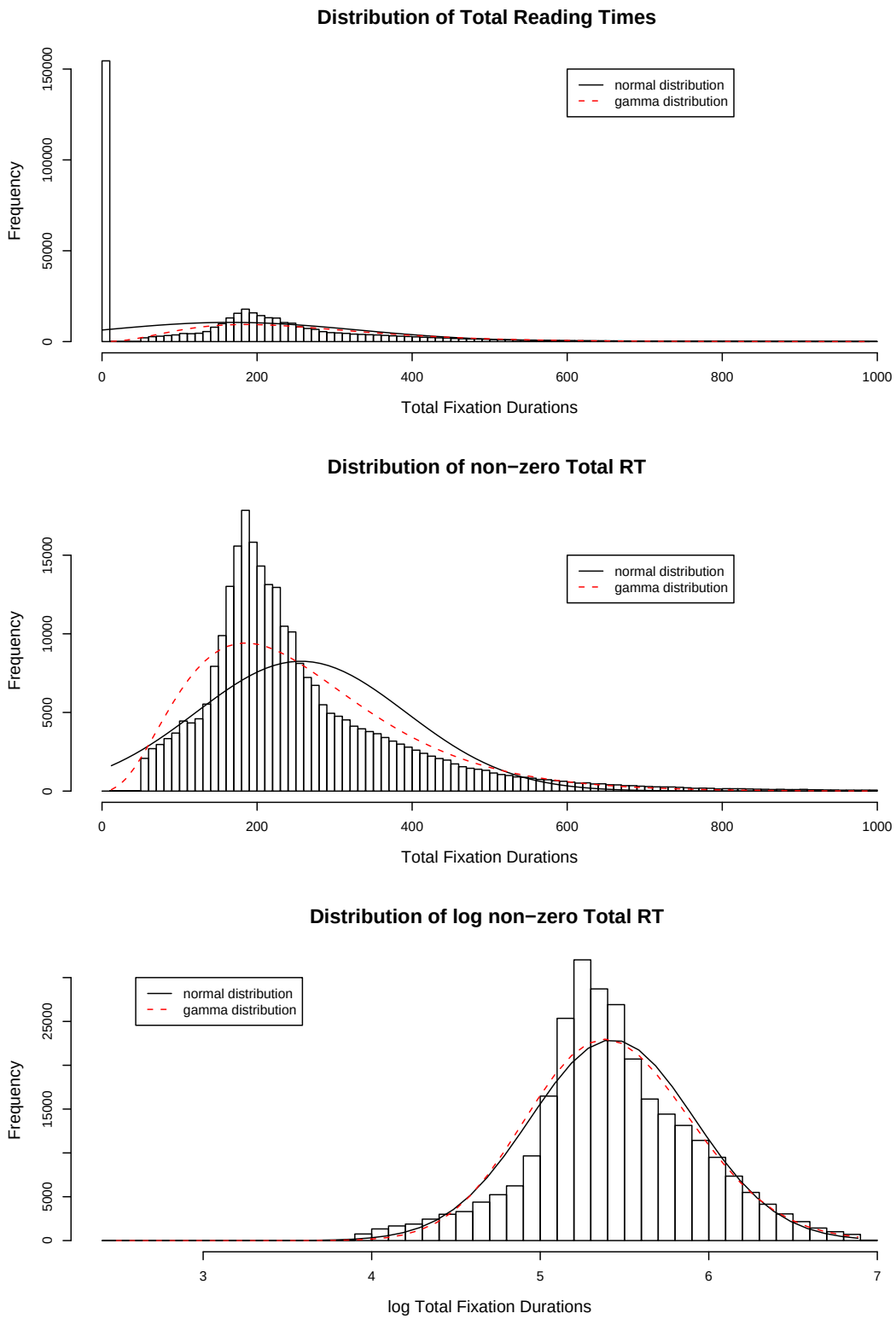


Figure 3.3: Distribution of Total Reading Times

3.1.1.2 Skipping, Refixations and Regressions

Skipping probability is 45% for first pass reading in the Dundee corpus. This rate is higher than previously reported – figures in the literature talk about a bit more than one third of the words being skipped in first pass reading (Brysbaert and Vitu, 1998). This difference can possibly be attributed to the newspaper text type, and possibly the eye-tracker and post-processing. Figure 3.4 shows a histogram of the number of fixations. Fixating exactly once is the most frequent event, with the probability of more fixations dropping quickly (the distribution is log-linear). Refixations as shown in the left-hand diagram comprise both multiple fixations on the same word during a single pass and regressions back onto a word.

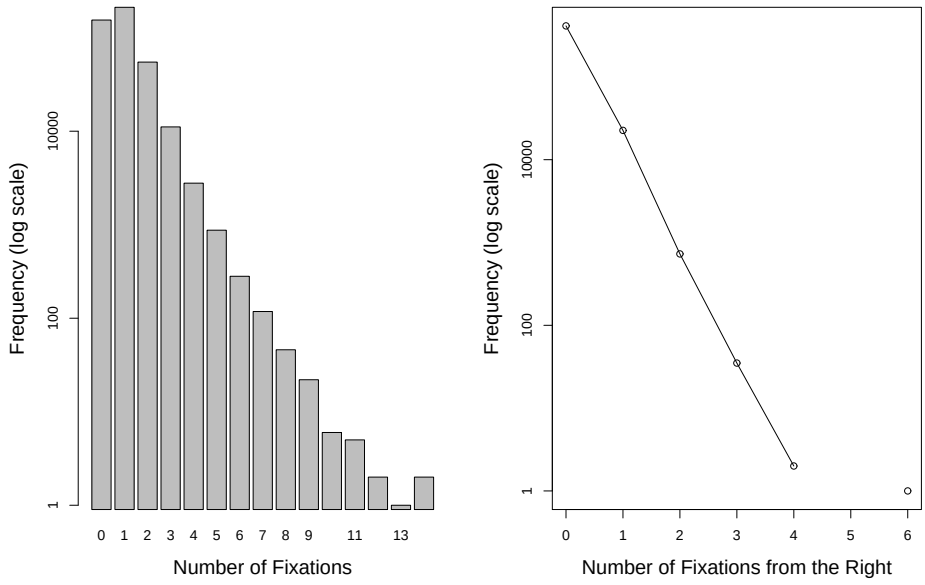


Figure 3.4: Distribution of Number of Fixations per Word and Number of Regressions onto a Word.

The probability of regressions out from a word in the Dundee Corpus is similar to the rates that have been previously reported in the literature, where people found that about 10% of saccades were directed to the left. In the Dundee Corpus, we find backward saccades in about 12.5% of all fixations, but these comprise leftward saccades that stay within the same word. The proportion of regressions leaving the word to the left is 10.8%, which corresponds to previously reported rates. This regression rate means that about 5% of all words (which are not at the end or beginning of a line) are the target of a regression, which corresponds to 8.3% of all fixated words. Figure

3.5 shows a histogram of regressions. The scale is log-linear, this means that most words are not source of a regression, about 22,000 words are the source of exactly one regression, 781 are the source of two regressions etc.

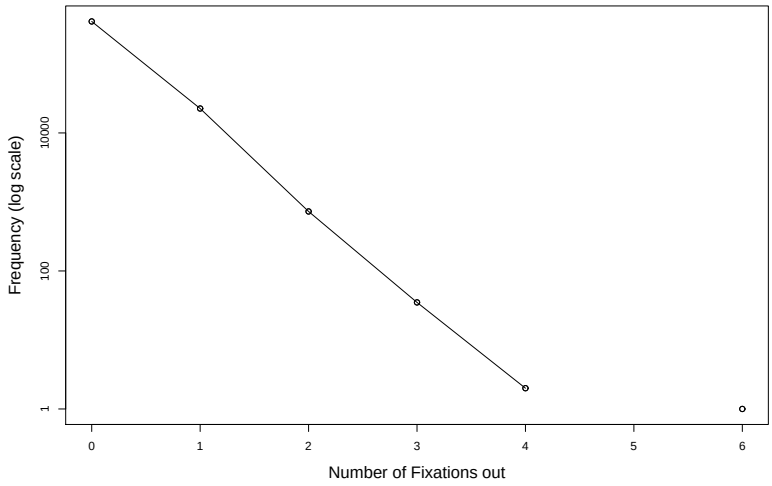


Figure 3.5: Distribution of Number of Regressions out per Word

3.1.1.3 Track Loss

The rate of track losses is unfortunately quite high in the corpus. We define a track loss as a sequence of four adjacent words that are not fixated. Out of the approx. half a million tracked words (50,000 words $\times$ 10 participants), 7.3% of the data points are invalid due to track loss. Regions of track loss are excluded from all regression analyses and statistics calculated for this thesis, since the large proportion of track loss risks to distort the data substantially, in particular for estimating skipping and refixation probabilities.

3.1.1.4 Inter-subject Variability

One disadvantage of the Dundee Corpus is that it was only read by 10 subjects. Figure 3.6 shows six box-and-whisker diagrams that display the differences in reading behaviour of the 10 subjects. The plots in the first column show the mean and variation in first fixation duration, first pass duration and total reading time for the 10 subjects; the second column shows a zoomed-in version of the corresponding plots in the first column. All of these plots refer to fixated words only. The height of the boxes indicates the first and third quartile of the data points. The length of the whiskers are

calculated from the inter-quartile-range (IRQ), which is the difference between the first and the third quartile. The whiskers are then defined as reaching 1.5×IRQ from the lower quartile and 1.5×IRQ from the upper quartile. Any points outside this span are traditionally regarded as outliers.

There is a substantial amount of variation both within and between subjects: Within-subject variance is evident through the large number of "outliers", i.e. points that are above or below the whiskers. Between-subject variance can be better seen in the zoomed-in versions of the plots in the right-hand column, the notches around the mean are quite tight and only partially overlap, which means that the subjects have different reading behaviours. (If the notches of two plots do not overlap this is strong evidence that the two medians differ (Chambers et al., 1983, p. 62).) For example, we can see from Figure 3.6 that subject "sg" exhibits more variation in first fixation durations than the other subjects, and subjects "sg" and "sc" have shortest fixation times, while the fixation times of subject "sb" are longest among the participants.

See also the left-hand side of Figure 3.7 for differences in participants' average skipping probability per word, number of fixations per word and the probability that a word in the corpus is the source of a regressions. The right-hand side figures depict first pass launch distances[3] (all data points and zoomed in around the means). We can see that subject "sg" (the one with short fixation times) skips words least often (only 3 out of 10 words are skipped) and has the highest fixation and regression rates (above 1 in 10 words) among the participants, while "se" skips 45% of words, and has a low fixation and regression rate. Reader "sa" and "se" make the longest saccades, which corresponds to the finding that they have the smallest number of average fixations on a word and largest skipping probabilities.

The observed differences in reading behaviours between subjects are expected. In practice, this illustrates why it is important in our studies to model subjects as a random effect in regression models.

[3] All launch distances have negative values, because only those from first pass reading are shown, which by definition only count as first pass if there has not yet been a fixation to the right of the word.

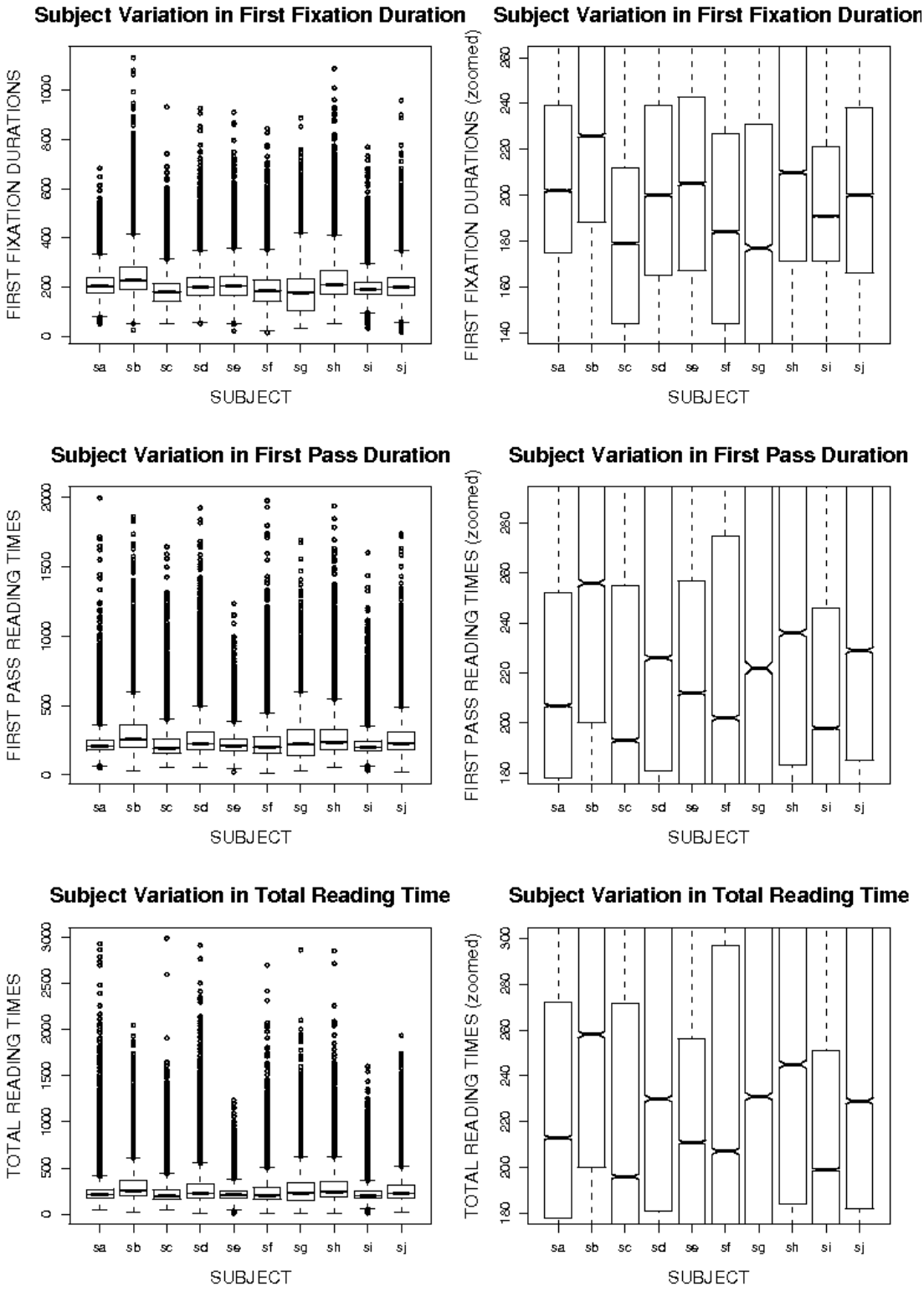


Figure 3.6: Variation in Reading Times across the 10 participants who read the Dundee Corpus.

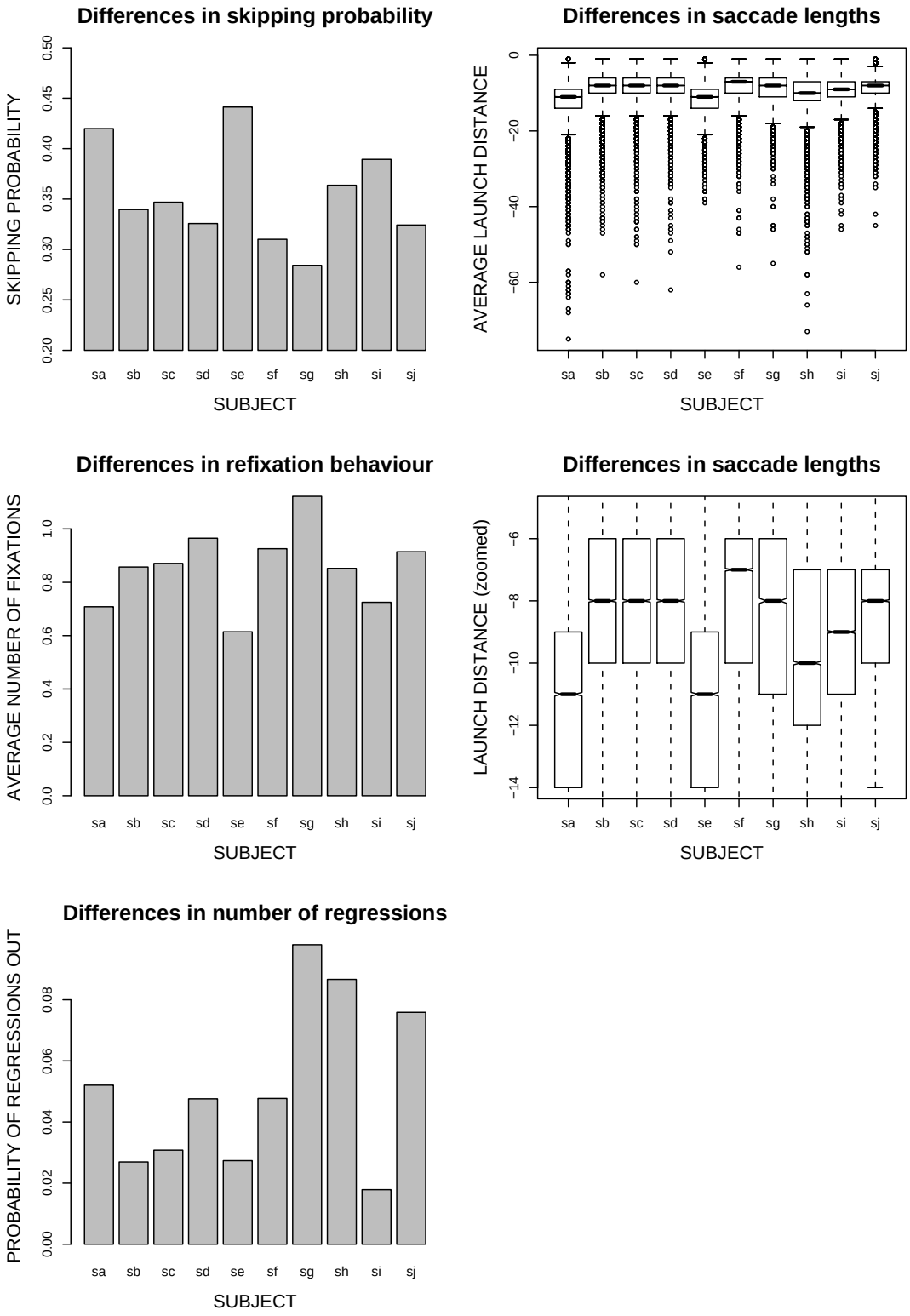


Figure 3.7: Variation in fixation behaviour across the 10 participants who read the Dundee Corpus.

3.1.2 Distributions for Low-Level Variables

The distribution of low-level variables which we discuss in this section follows the expected distributions which have also been found for other eye-tracking experiments.

3.1.2.1 Fixation Landing Position and Launch Distance

Figure 3.8 shows the distributions of fixation landing positions and launch distances in the corpus. The plotted fixation landing positions are not normalised for word length here, hence the strong skew. The small bar at landing position -1 is due to the fact that this plot shows landing positions on words and not on what's called an object in the terminology of the corpus: An object is a word plus its punctuation. Thus, it can happen that the fixation lands e.g. on quotes before a word, but since there is no space in-between the word and the punctuation, they are counted as one object, leading to possible negative fixation positions for words.

Launch distance is the distance from the current landing position to the preceding one. Launch distances have a peak at about -8 characters, which is the median launch distance for most subjects (see Figure 3.7). The unexpectedly high number of launch distances with length zero is an artefact in the data: the first fixation on a new screen is assigned launch distance 0 in the Dundee Corpus. The distribution of launch distances furthermore also exhibits a skew to the left. This can be explained by the fact that words toward the beginning of the line cannot possibly have launch distances beyond the number of characters between them and the beginning of the line.

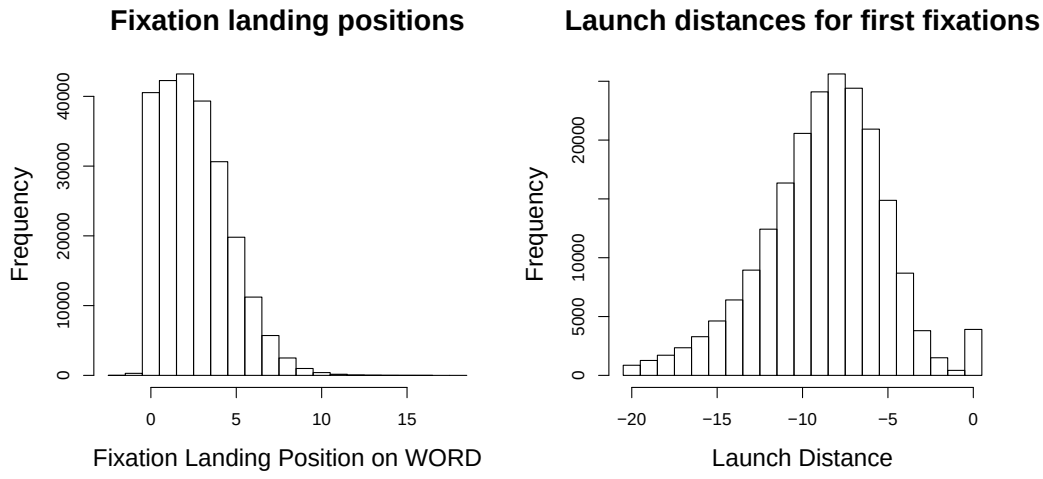


Figure 3.8: The distributions of fixation landing position on a word and launch distances in the Dundee Corpus.

3.1.2.2 The IOVP effect

A well-known effect is the inverse optimal viewing position (IOVP) effect (Vitu et al., 2001), which refers to the fact that fixations at the beginning or end of a word are shorter than fixations on the middle of a word. In the Dundee Corpus, we also find clear evidence of the IOVP effect, i.e. that fixation durations are longer when the fixation lands at the middle of a word. Figure 3.9 plots the fixation durations against the landing positions, conditioned on the length of the word, that is all words with a certain length are grouped together. For example, the plot in the first column of the third row in the figure shows fixation durations for all words of length 8 (the bottom left plot shows all words of length 1, the one on its right all words of length 2 and so on). Fixation durations were longer when the fixation landed on the 4th or 5th character of the word than when they landed on the first or last character of the word. This pattern is pretty stable for words up to length 13. After that, the pattern becomes less regular due to the low number of observations.

Interestingly, variance seems to be pretty much constant across the fixation positions, thus not supporting the hypothesis that the IOVP effect would be an artefact due to fixations on the beginning and end of a word being either very long (because it is difficult to see the word) or very short because of immediate refixations to a more optimal position, which was proposed by Engbert et al. (2005).

3.1.2.3 Word Length

Word length is an established influencing factor of fixation durations. The longer the word, the longer the fixations, and the smaller the probability that a word is skipped. The distribution of word lengths in the Dundee corpus is shown in the first plot in Figure 3.10. The plot below it shows the average numbers of fixations for the different word lengths. The number of fixations increases linearly with word length. Skipping probability decreases exponentially with increasing word length, as depicted in the bottom left plot. Words with more than 15 characters are virtually never skipped; and even for words with more than 6 letters, skipping probability falls below 10%. The plots in the right column of Figure 3.10 show the main effect of word length on reading time. There is a almost no effect in first fixation times, and a very large effect in first pass reading time and total reading time, which can be explained by the linear increase of refixation probability with increasing length. The plots also show that variance in fixation durations increases substantially with word length.

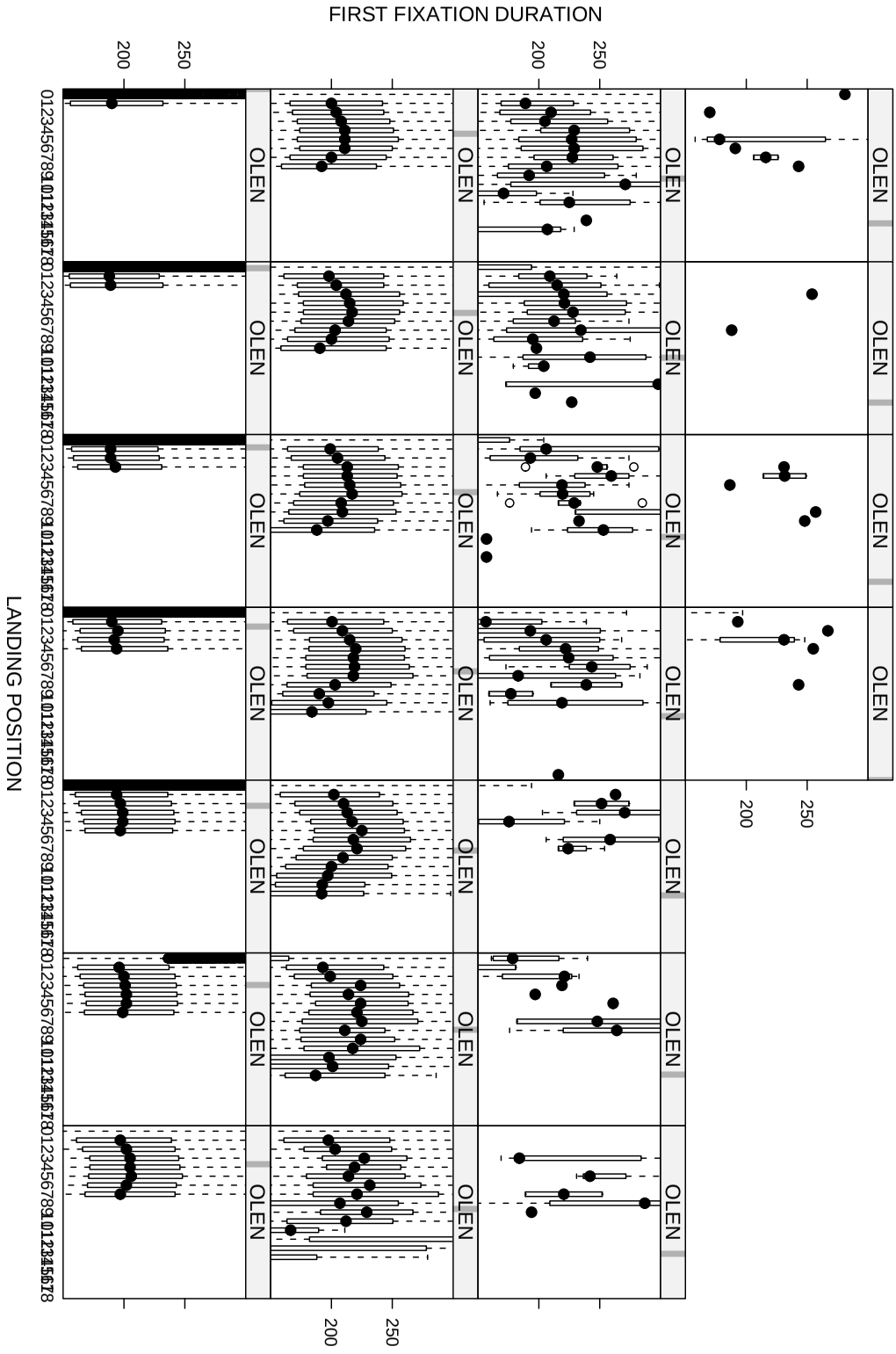


Figure 3.9: The IOVP effect by word length including punctuation (marked as “OLEN” in this figure).

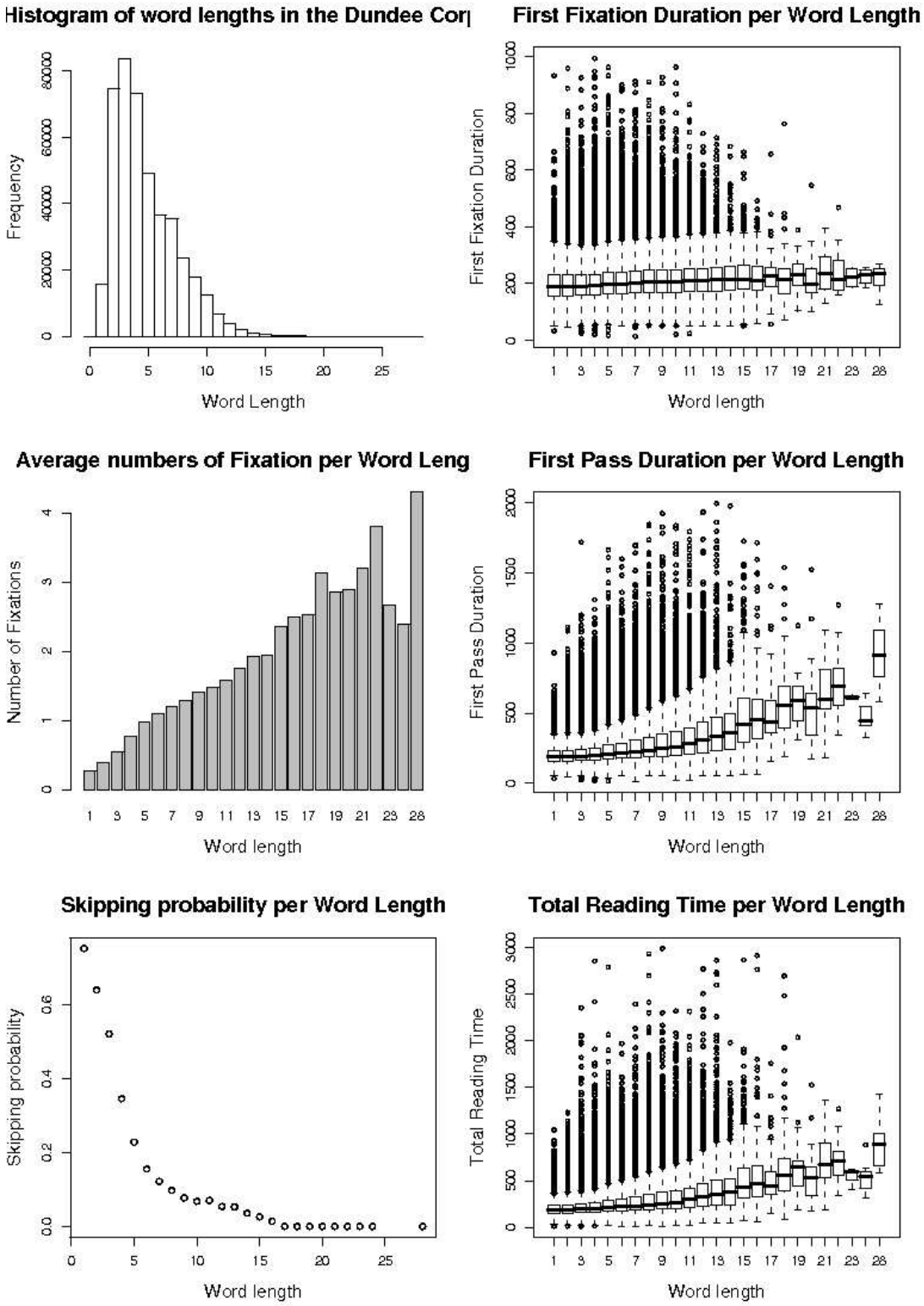


Figure 3.10: The influence of word length.

3.1.2.4 Word Frequency

The frequency of a word also has a strong influence on fixation durations. The more frequent a word, the shorter the fixation durations. In this work, we use frequency estimates from the British National Corpus (BNC), after stripping off punctuation. Their distribution is shown in the first histogram in Figure 3.11. The distribution is zipfian and follows expectations: there are some very frequent words, many frequent words and a long tail of infrequent words. Note the unexpectedly high number of words with $\log_{10}$ frequency per million words smaller than -0.5, which we are going to have a closer look at below.

The second histogram in Figure 3.11 shows the distribution of the frequency of words in the corpus based on frequencies in the Dundee corpus itself, it can thus be regarded as a local text frequency. Importantly, this distribution looks very different from the BNC-based estimate, hugely overestimating the proportion of rare words. Therefore it is important to estimate frequencies from a larger resource.

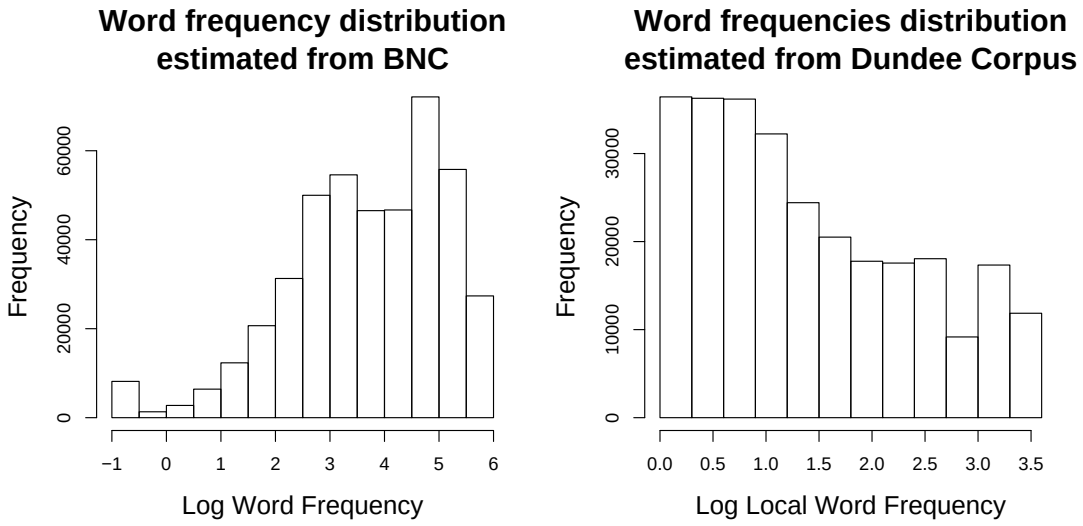


Figure 3.11: The distributions of frequency estimated from the British National Corpus and the Dundee Corpus itself. Log word frequencies in from the BNC were normalised for occurrences in a million and log transformed with $\log_{10}$. The local frequencies are also $\log_{10}$ transformed, but are scaled for number of occurrences against total number of words in the corpus.

Coming back to the unexpectedly high number of very rare words in the left subfigure of Figure 3.11, there are two possible explanations for the large number of very infrequent words. Firstly, some of the words in this class may have been assigned

inappropriately low frequencies. This can for example be the case for numbers and compounds. A second possibility is that these are words which are specific to the topics of the text that they occur in, for example if the newspaper article talks about some very rare species of animal, or used some acronyms that are introduced in the text and would not occur in the BNC. The first case is a more severe problem, as the second problem could be fixed by also using local text frequency as a predictor in regressions.

Figure 3.12 gives some insight into the first problem category. In the top left hand plot, we can see that the variation in word length is very large for words in the lowest frequency bin. We would normally expect a monotone relationship between word length and word frequency, with frequent words being shorter than infrequent words. Similarly, infrequent words from the most infrequent class should be skipped least often and receive most fixations. But this is not the case, as the second and third plot in the left column show. So let's try what happens if we exclude from the analysis all words that contain digits, special symbols (like '$', hyphens) or contain several capital letters. The variation in word length of rare words decreases considerably, and both skipping probability and fixation numbers become monotonous functions, with the rare words skipped least often and fixated (and regressed to) most often (see plots in the right column of Figure 3.12).

As can be seen in plots 3.13(a) and (b), leaving these data points out also has the corresponding effect on the distribution of reading times. When average fixation durations are plotted for each word frequency class, rare words on average receive shorter fixations than would be expected given their frequency (see top row plots in subfigures (a) and (b) of Figure 3.13). This effect is removed when words with digits or special characters and abbreviations are removed from the data set. The question is then how to handle these data points. On the one hand, they could be left in and possibly be explained by an interaction between word frequency and word length. The other solution is to either leave them out of the regression analyses, or to change their frequency assignment. For instance, a psycholinguistic reason for changing the frequency assignment of digits would be that they are probably considered as a class of signs in the human processor and therefore should be annotated with their class frequency. Compounds with hyphens on the other hand should not be annotated with the frequency for the whole compound, as there is evidence in the literature on compound reading that the reading durations of compounds are primarily dependent on the frequency of the first part of the compound (Juhasz et al. (2003)). The studies reported in this thesis exclude words that contain digits, special characters or several upper case letters.

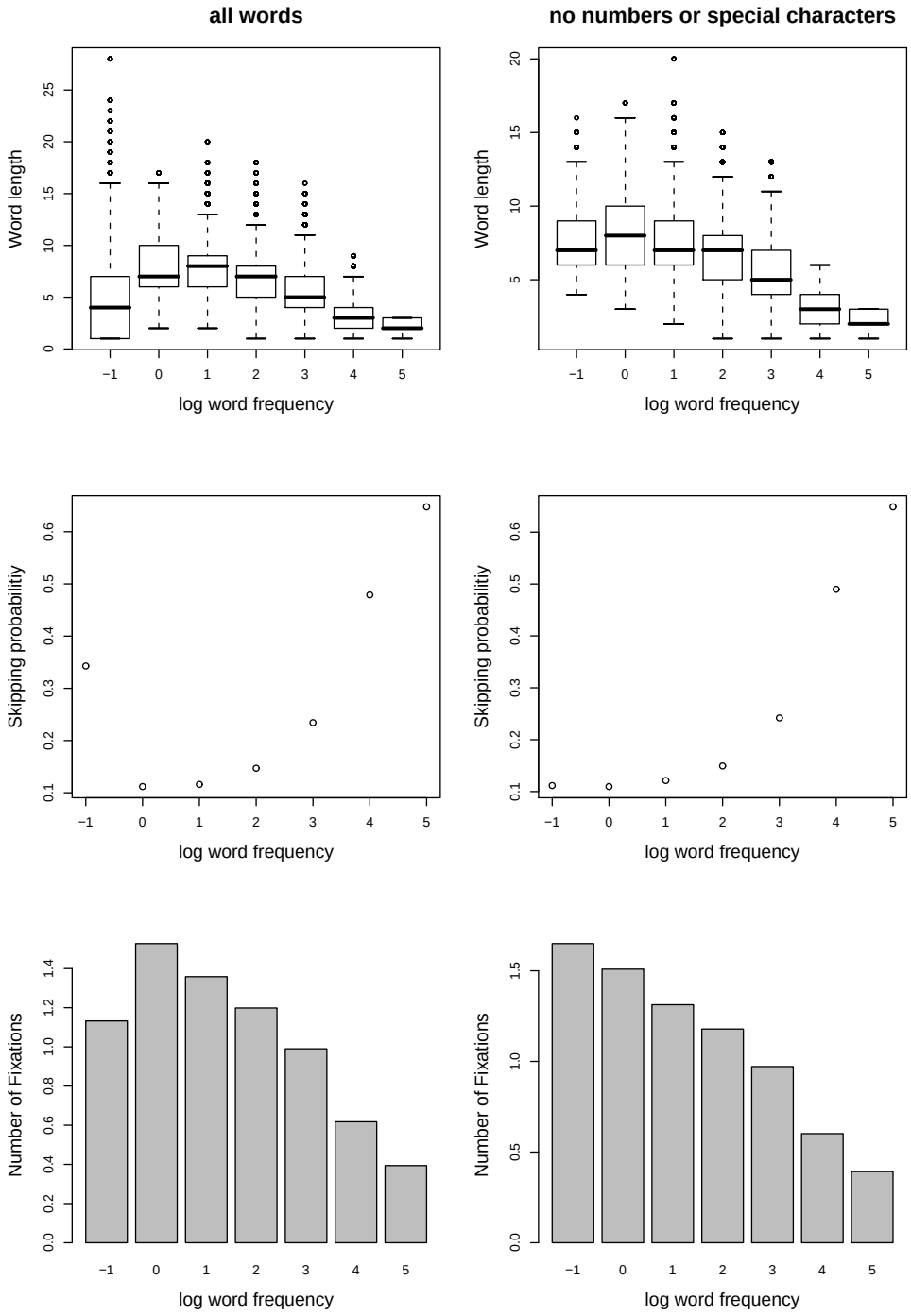
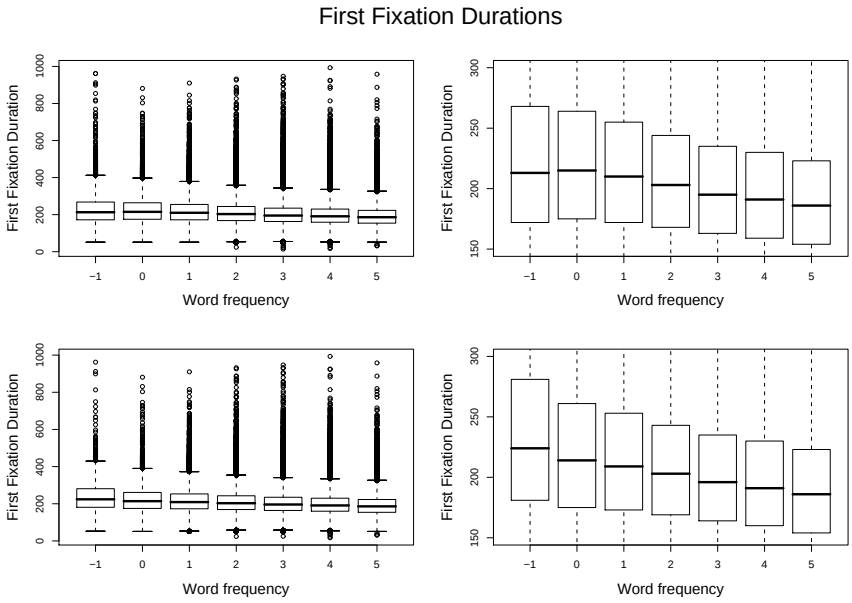
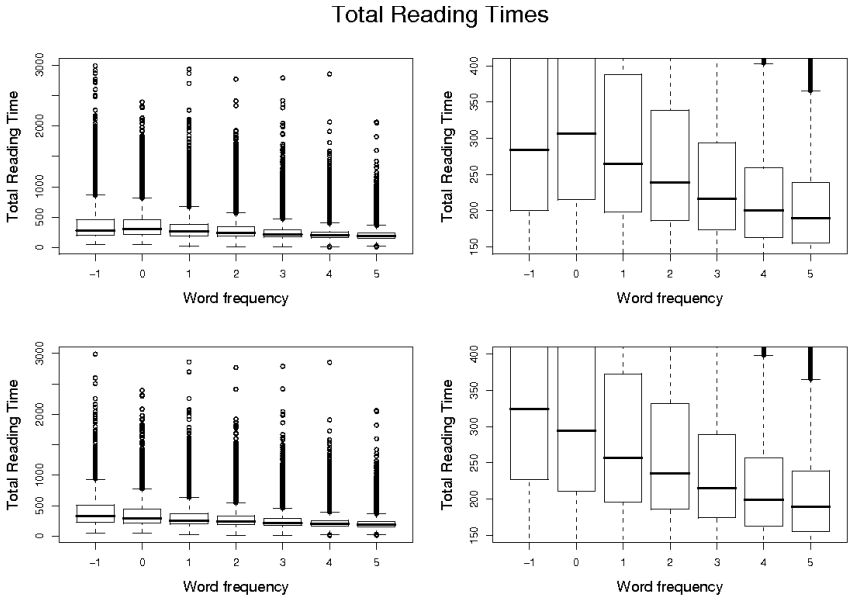


Figure 3.12: The first column shows word length distributions, skipping probability and numbers of fixation on a word for words of different frequency classes. The second column matches the plots from the first column, but the data set of the second column excludes all words with symbols that are not characters, such as numbers, punctuation, compounds with a hyphen or special signs.



(a) Distributions for first fixation durations by word frequency.



(b) Distributions for total reading times by word frequency.

Figure 3.13: The distributions of reading times for different word frequencies. Plots in the first row show distributions the complete data set, while plots in the second row exclude all words containing digits, special symbols or several capital letters. The plots in the second column zoom in on the region around the median durations of the data from the corresponding left column plots.

Figures 3.13(a) and (b) also show the main effect of frequency on the reading measures, which confirms the expectation that more frequent words are read faster than less frequent ones, in particular when only considering words which contain nothing but characters and no special signs.

3.1.2.5 Transitional Probabilities

The forward transitional probability of a word is the conditional probability of word w_n given word w_{n-1}. We estimated these probabilities from the BNC using the CMU-Cambridge Statistical Language Modelling Toolkit (Clarkson and Rosenfeld, 1997). Transitional probabilities have been shown to influence fixation durations. There are two reasons for why forward transitional probabilities can be expected to influence reading: the first reason is rather low-level, saying that two words with high transitional probability look visually familiar and are therefore easy to read because they often occur together. The second reason is that these word bigrams actually capture the predictability of a word given the last word, and thus also capture some of the linguistic structure.

The first plot of Figure 3.14 shows the distribution of log forward transitional probabilities (FTP) in the corpus. FTPs were estimated from the BNC, after stripping any punctuation (his makes the distribution much smoother and helps to alleviate data sparseness problems). The top right plot shows how FTPs are correlated with the number of fixations on a word. The relationship is log-linear with the number of fixations increasing the less predictable a word is given the previous word. The main effect of transitional probabilities can be read from the bottom four plots in Figure 3.14, indicating that reading times are the longer for less predictable words.

It is also informative to look at the interaction between unigram frequencies and forward transitional probabilities, shown in Figure 3.15. The relationship is very strong and mainly linear, with words with higher frequency also having high transitional probabilities. An exception is the cloud of points with transitional probabilities between 0 and -2.5, where the corresponding unigram probabilities seem to be distributed all over the place, instead of having high frequencies. In particular, almost all words with a unigram probability smaller than zero seem to be in this cloud. These are cases where the current word has not been seen (or has not been seen often often enough; there's a frequency cut-off of 65,000 words for the vocabulary size) in the corpus. For such cases, smoothing (i.e. some of the probability mass is taken away from observed events and reserved for unseen events) is applied: we use the transitional probability of see-

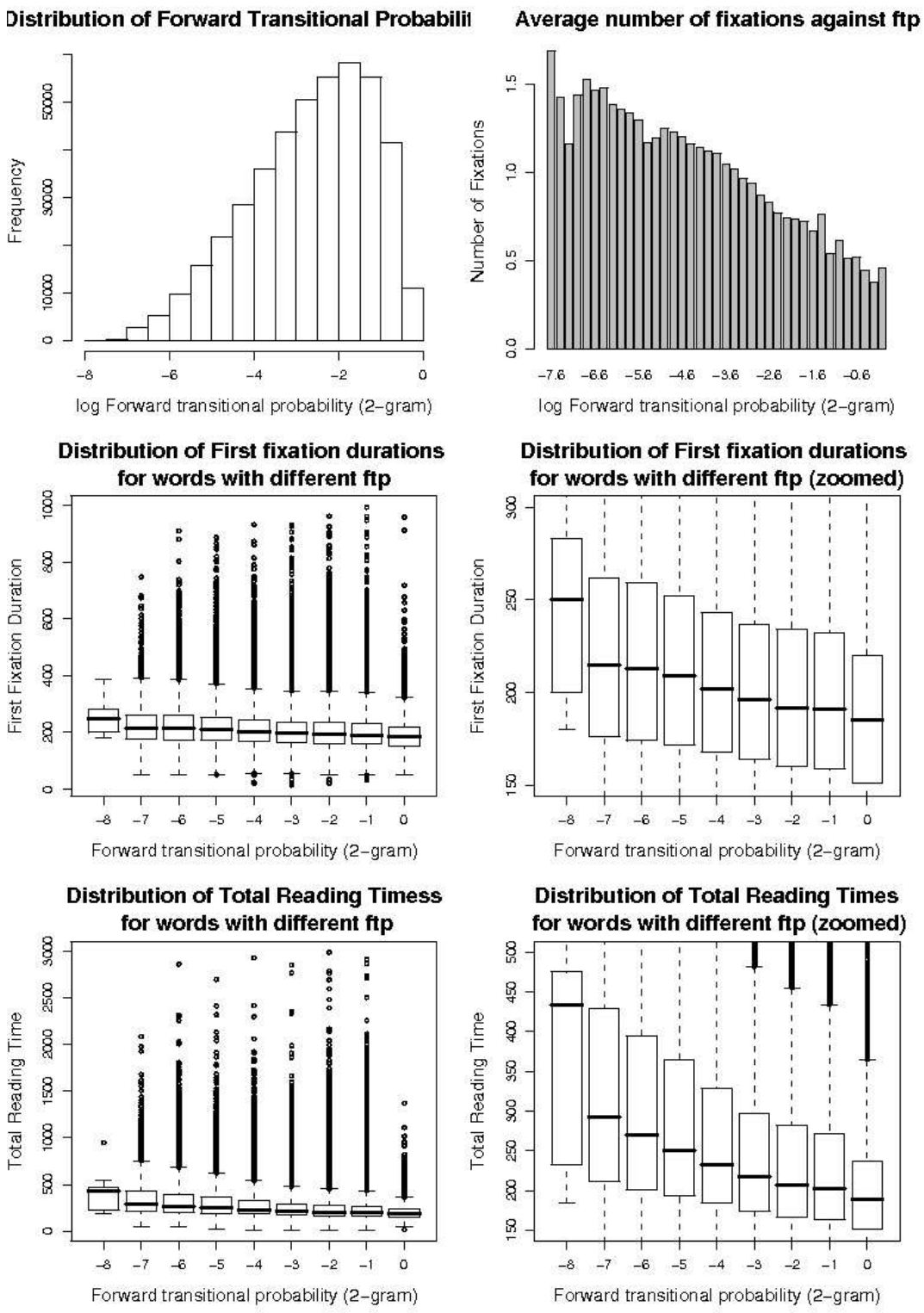


Figure 3.14: Distribution of forward transitional probabilities in the corpus and their main effect on reading measures.

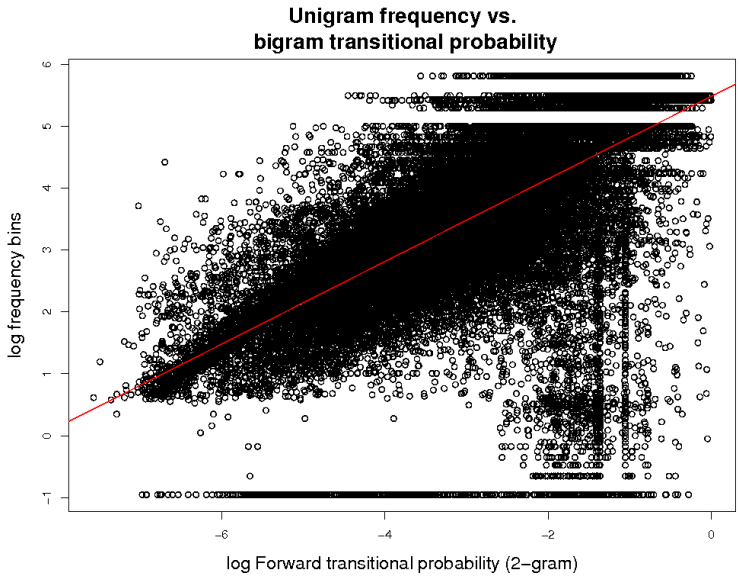


Figure 3.15: This plot shows the correlation between log frequencies and log transitional probabilities for the Dundee Corpus.

ing a generic “unknown” word given the previous word (estimated from replacing rare words in the training corpus by a label “unknown”). With this practice, the probability of seeing an unknown word following the word “Mr” is for example much more likely than seeing an unknown word following “but”. There are of course also cases where both the current and the previous word are unknown. The smoothed log probability for these cases is -1.05 (again, this was estimated from the training corpus by replacing rare words with the “unknown” label); in the plot this corresponds to the right hand dotted line at -1.05 that is parallel to the y-axis.

The horizontal line of points at frequency -1 corresponds to unknown words, since they are assigned this value as their smoothed probability. These words still have different FTPs because even when a cut-off occurs in the unigram frequency estimation, it may not be below the cut-off threshold for the bigram estimation. Commonly occurring examples for such cases are the estimation of digits following the word *around*. Finally, the horizontal clusters of dots with identical frequencies between log frequency values of 5 and 6 are due to multiple occurrences of common words, whose transitional probability differs according to their context.

We also calculated backward transitional probabilities (BTP) from the BNC. BTPs estimate the probability of the current word given the following word. The idea here is that backward transitional probabilities are thought to capture preview effects. The distribution is much more peaked than the distribution for forward transitional prob-

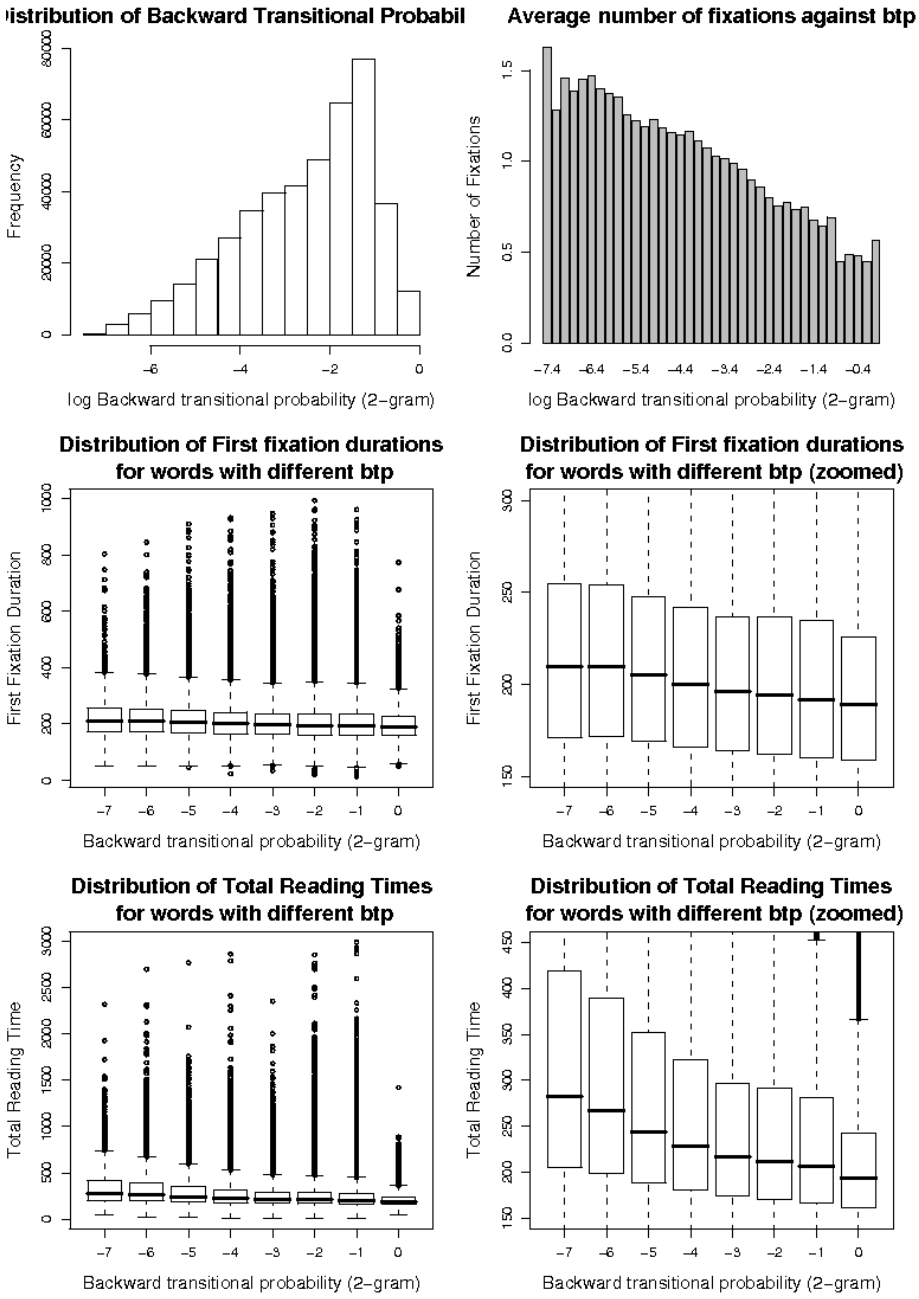


Figure 3.16: Distribution of backward transitional probabilities in the corpus and their main effect on reading measures.

abilities, but shows similar behaviour, with the number of fixations increasing log-linearly with decreasing transitional probability, and both first fixation durations and total reading times decreasing with increasing transitional probability. Overall, the backward transitional probability effect seems to be smaller than the forward transitional probability effect (see Figure 3.16).

3.1.3 Distribution of Explanatory Variables for Syntactic Processing

This section gives a brief overview of the distribution of the higher-level explanatory variables in the corpus. There are no full regressions in this section, so effects may be disguised or covered by other factors.

3.1.3.1 Surprisal

Surprisal was proposed by Hale (2001) as a measure of syntactic sentence processing difficulty. The Surprisal of a word in a sentential context corresponds to the probability mass of the analyses that are not consistent with the new word. For a detailed description of Surprisal, see Section 2.2.4 in Chapter 2. Two different versions of Surprisal were analysed: lexical Surprisal and structural (or unlexicalized) Surprisal. lexical Surprisal takes into account the probabilities of the grammar rules for non-terminals as well as the probabilities for terminals, i.e. the probabilities of a word given a POS-tag. It therefore also captures aspects quite similar to word frequency. Structural Surprisal on the other hand only takes into account the probabilities of the rules involving non-terminals.

Lexical Surprisal The top right-hand plot in Figure 3.17 shows a histogram for lexical Surprisal as calculated using the Roark parser. The distribution is similar to the one for forward transitional probability (but mirrored because Surprisal uses negative log values). Furthermore, there seems to be a correlation between lexicalized Surprisal and reading measures: More surprising words are fixated more often, and are fixated for longer according to both first fixation duration, first pass duration (the latter is not shown in Figure 3.17) and total reading time. However, this effect might be confounded with simple lexical frequencies. Therefore, it is necessary to run a multiple regression in order to factor out these effects, and find out whether Surprisal values contribute anything to explaining the data above and beyond simple frequencies.

Structural Surprisal Figures 3.18 and 3.19 show the distribution of two versions of structural Surprisal from the Roark parser, and their correlation with reading measures. The data in Figure 3.18 is calculated the same way as lexical Surprisal, but the lexical probabilities are subtracted, in order to eliminate lexical frequency effects. There still seems to be a positive correlation between reading measures and Surprisal, since the

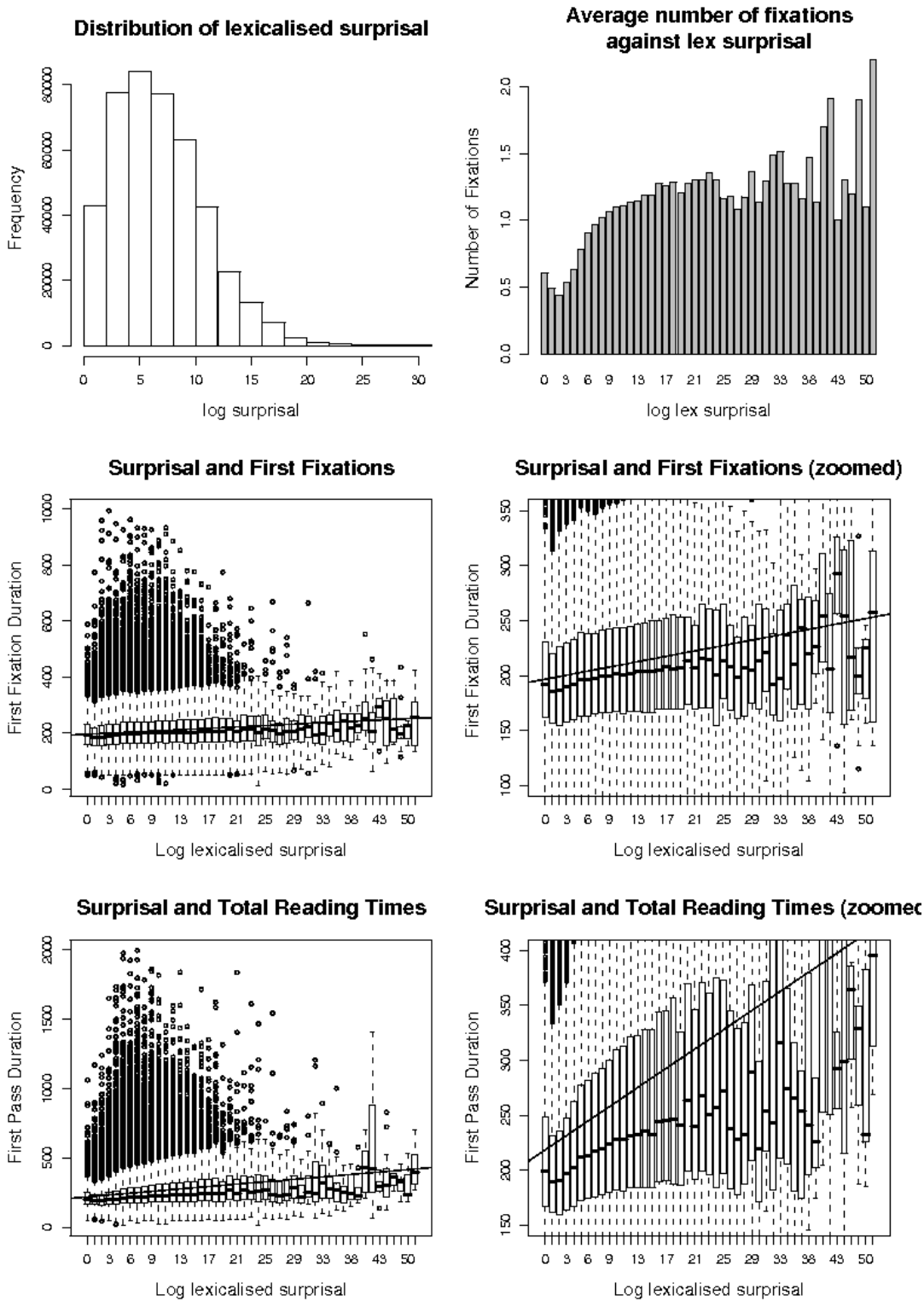


Figure 3.17: The distribution and main effect on reading measures of lexical Surprisal as calculated using the Roark parser.

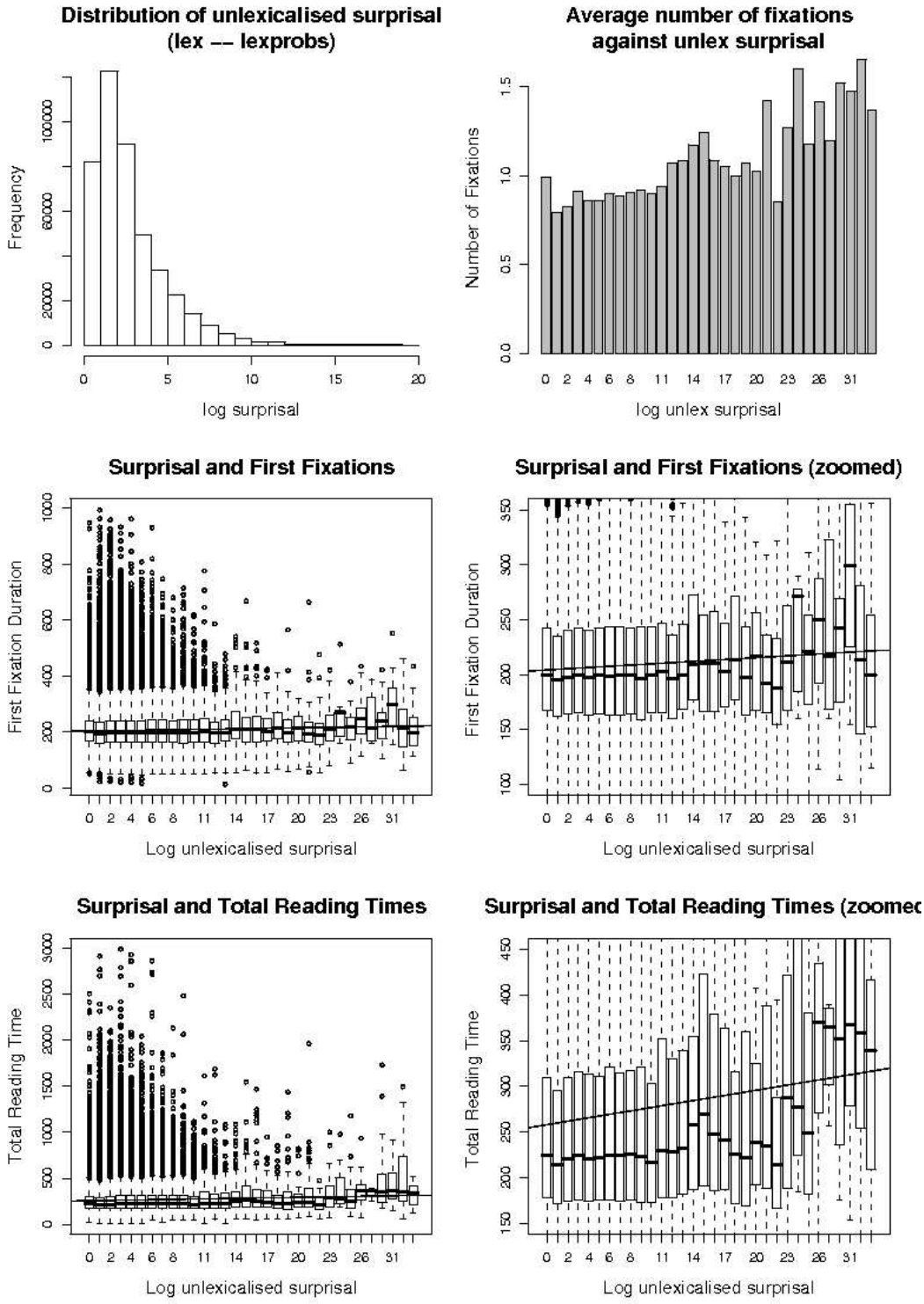


Figure 3.18: The distribution and main effect on reading measures of structural Surprisal as calculated using the Roark parser, calculating lexicalized probabilities and subtracting lexical probabilities.

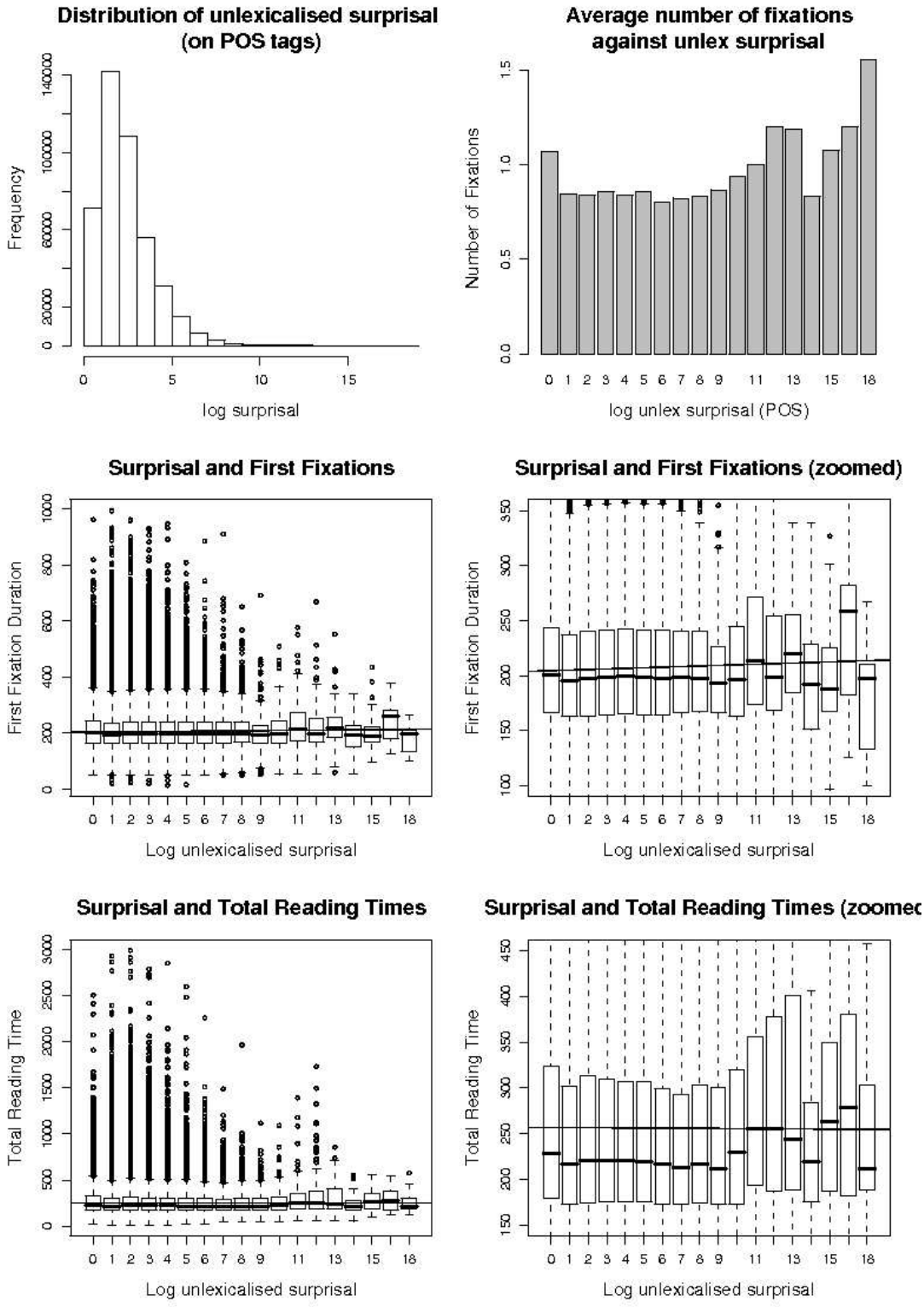


Figure 3.19: The distribution and main effect on reading measures of structural Surprisal as calculated using the Roark parser, calculation based on POS tag sequences.

regression lines have a positive gradient.

However, there is no visible effect for the second version of structural Surprisal. This second version differs from the first version in that the probabilities are estimated from POS-tag sequences. This means that all information about subcategorization frames is ignored. This second version seems to capture less of an effect, if anything, the regression line in the bottom right plot from Figure 3.19 seems to be descending, which would mean that words would be read faster when they were more surprising. Note though that little can be derived from such a simple correlation, since none of the potentially confounding effects have been filtered out.

In our regression models, structural Surprisal using the first method turned out to be a better predictor of reading times, which is why all future mentions of structural surprisal in this thesis refer to the first version. This result is corroborated by Roark et al. (2009), who, on a different corpus, found an effect of structural Surprisal using the first method, but no effect using the second method.

3.1.3.2 Dependency Locality Theory

Another theory for processing difficulty, Dependency Locality Theory (DLT), was proposed by Gibson (1998, 2000). 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. Please refer to Chapter 2, Section 2.2.2 for a detailed account of DLT and its two components, integration cost and storage cost. Note that in our analysis here, we only show plots for integration cost, because we only use this component in later experiments. The cause for this is partially that in Gibson (1998), only the integration cost component is used as an approximation to DLT, and partially that we did not find storage cost to be a significant predictor of reading times.

The distribution for integration costs is shown in Figure 3.20. It looks quite different from the Surprisal distributions: There is a large number of words with an integration cost equal to 1, and the number of words with higher integration cost drops log-linearly. There is no clear correlation with the number of fixations, but a positive gradient for the regression line for first fixation durations. Again, these results have to be analysed more carefully while taking into account potentially confounding factors.

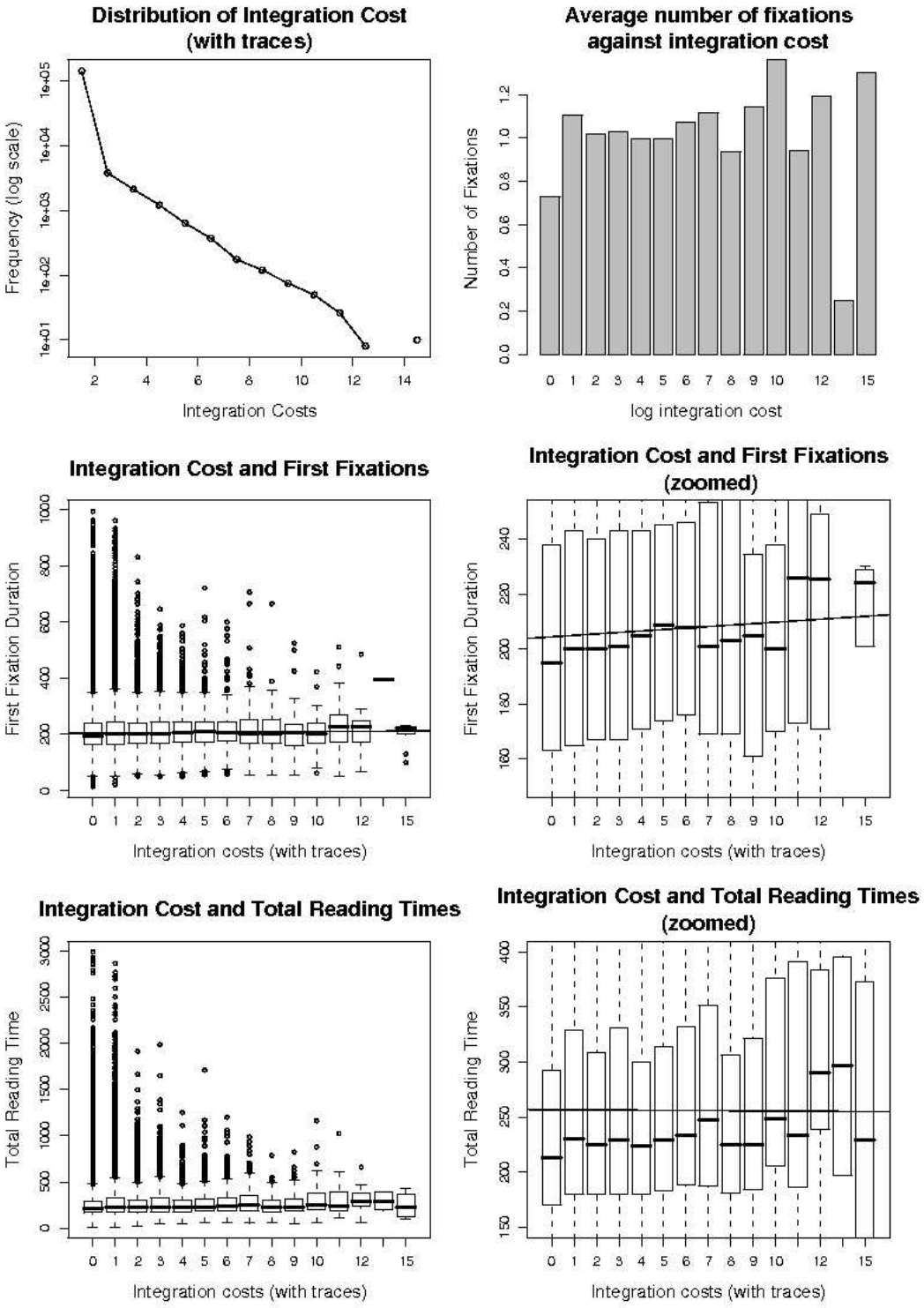


Figure 3.20: The distribution and correlation with reading measures of DLT integration cost based on the dependency parses from the MINIPAR parser.

3.1.4 Discussion

The main function of this section was to provide an overview of the characteristics of the Dundee Corpus and introduce the main factors that are known to influence reading times. We saw that the low-level factors behave as we would expect after what we know from other eye-tracking studies, with typical distributions for launch distances and landing positions in a word, the IOVP effect and with longer word length and lower frequency corresponding to longer reading times. An important observation from analysing frequencies was the influence of digits and acronyms. Such data points do not usually occur in eye-tracking experiments since experimental materials are usually purpose-designed and there is no reason for including such items. While we could see quite strong correlations between the low-level variables and reading times, such correlations were not as strong for the higher-level syntactic predictors. This can be considered as a first indication that the explanatory power of syntactic effects on reading time in naturally occurring data is not as strong as the influences from more low-level variables. Because reading times are influenced by many factors, some of which have a large impact on fixation durations, it is important to account for these low-level effects before trying to find correlations between more subtle or complex effects and reading times. The following section will discuss linear mixed-effects models for analysing the Dundee corpus and finding out whether the variables we are interested in have any explanatory power for the reading times.

3.2 Method: Mixed-Effects Models

There are two types of mixed effects models which we will discuss here: hierarchical linear mixed effects models (Pinheiro and Bates, 2000), as well as mixed effects models with crossed random effects (Baayen et al., 2008). Both are a generalisation of linear regression that allows the inclusion of random factors (such as participants or items) as well as fixed factors, hence the name “mixed” effect models. The fixed factors can be discrete (such as whether the previous word was fixated) or continuous (such as word frequency).

This section first motivates the use of mixed-effects models in this work, and then discusses which specifications within mixed effects models should be used to model the data best.

3.2.1 Regression Analysis

In general, regression analysis refers to modelling a response variable y (in our case, the fixation durations) as a function of one or more explanatory variables $x_1..x_n$ (in our case, length, frequency, landing position, Surprisal value, etc.). In the regression, an intercept i and one regression coefficient $\beta_1..\beta_n$ for each of the explanatory variables is estimated such that the best possible fit with the response variable is achieved. The remaining unexplained variance in the response variable is the error ε.

$$y = i + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \varepsilon$$

Once the intercept and regression coefficients for the explanatory variables have been estimated, one can be interested in the size of the error, the inverse of which tells us how much of the data (i.e. which proportion of the variance in the reading times) can be explained by the explanatory variables. We are here however mainly interested in whether the explanatory variables we are focusing on (i.e. the syntactic predictors) are able to explain any of the data above and beyond what can be explained by the more low-level explanatory variables. That is, we are looking at whether a regression coefficient that is found during the regression process is significantly different from zero, and whether it has the expected polarity (which tells us whether the relationship between the explanatory variable and the response variable are changing proportionally or anti-proportionally).

Assumptions for standard regression analyses include:

1. The response variable is normally distributed.
2. The variance of the error is constant across observations (homoscedasticity).
3. The independent variables are error-free.
4. The predictors are linearly independent, i.e. it must not be possible to express any predictor as a linear combination of the others.
5. The errors are uncorrelated, that is, the variance-covariance matrix of the errors is diagonal and each non-zero element is the variance of the error.

Not all of these basic assumptions are fulfilled by the raw reading time data. We will therefore discuss problematic aspects in the next sections. First, we will look at how a more normal distribution of the response variable can be achieved. We will

see that the models suffer a little bit from heteroscedasticity, but that this problem seems much less of an issue once the distribution of the response variable is close to a normal distribution. Regarding these two assumptions, Jacqmin-Gadda et al. (2007) argue that mixed effects models are quite robust to violations of these assumptions. The third assumption, error-freeness of the independent variables, i.e. that the values for length, frequency, Surprisal etc. are correct, is not necessarily true either. For some of the variables, like length, this is trivial, but frequency estimations depend on the corpus used and the estimates of processing difficulty such as Surprisal depend on parses from an automatic parser, which will be incorrect a sizable proportion of the time. However, we can't do anything against this problem – all the estimates are as good as possible given our tools. In addition, not all predictors are necessarily linearly independent of one another. This holds in particular for the more complex syntactic explanatory variables which may also capture more low-level effects and therefore not be independent of them. This problem, and how to deal with it, will be discussed in Section 3.2.3. Finally, we will review different ways of constructing the regression model, and discuss model selection and outlier removal.

3.2.2 Normal Distribution of the Response Variable

As seen in Figures 3.1 to 3.3, the response variable, reading time, is not exactly normally distributed but skew to the right. This non-normality violates the first assumption underlying the regression model. A more normal distribution of data points can be achieved by excluding all data points with zero fixation duration and log-transforming the reading times.

For the Dundee corpus, the skipping rate is approximately 45% for first fixations (i.e. 45% of the words are not fixated at first pass reading), and 35% of the words are never fixated. This means that zero reading times make up a considerable amount of the data, and therefore have an important influence on regression coefficients. If not treated separately, these data points increase residual variance in reading time regressions immensely. If one wants to include all data points into an analysis, it would be better to use the number of fixations as a response variable, or simply a flag, indicating whether a word has been fixated or not, and use a logistic regression model.

One way to try to overcome the problems that are due to non-normality of the reading time data is to use mathematically transformed reading times instead of raw reading times in the regression. Logarithmically transformed reading times are more

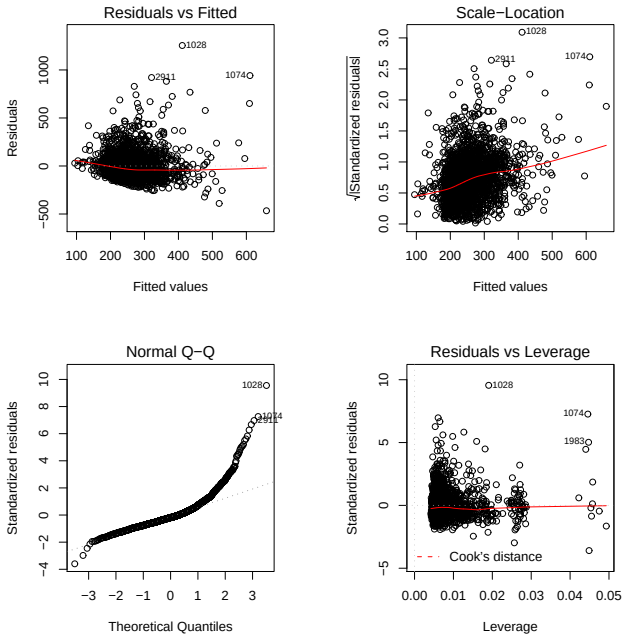
similar to a normal distribution than the raw values for many reading time data sets. For an example, recall Figures 3.1 to 3.3: the histograms show that the reading time data for all three reading time measures discussed here fit a normal distribution better when they are logarithmically transformed.

Figures 3.21 a) and b) show the error plots for raw and logarithmically transformed models for regressing total reading times. In the log model, heteroscedasticity occurs much less than in the raw reading times model (this can be seen from the shapes of the dots in the first plot of the two figures: While the residuals become larger as fitted values increase in the plot of raw reading times, there is no such pattern in the log reading times plot). In the quantile-quantile plots (bottom left plots of the two subfigures) we can see that the deviation from the linear line became much smaller (i.e. the data is less skew).

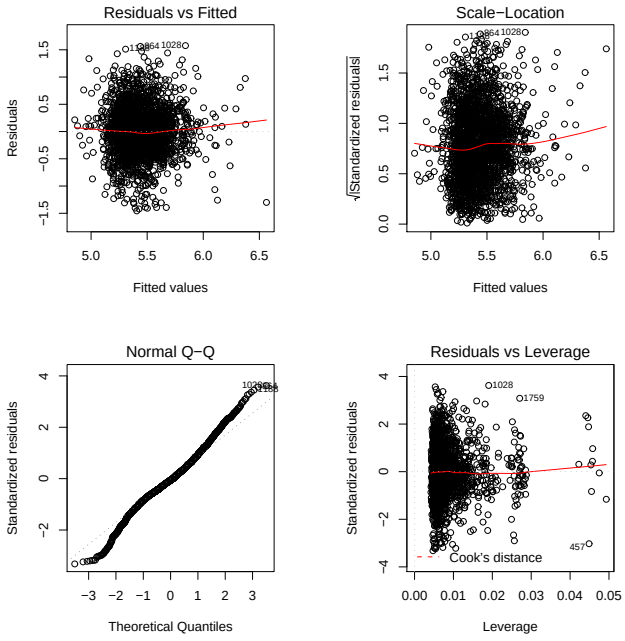
All of these arguments support the use of log-transformed reading times in regression models. A disadvantage with this practice is however that model results may be harder to interpret when the values of the response variable are transformed, which is harder to justify given claims that mixed-effects models are robust with respect to violation of normality. Due to this inconclusiveness, we always ran regressions with both the raw and the logarithmically transformed values. Generally, these models lead to the same conclusions. We will therefore report results with the raw reading time data, for the sake of interpretability. In the literature, people often use raw reading times and do not transform them logarithmically (logarithmic transformation of explanatory variables like transitional probabilities or frequencies, however, is very common). It seems to be generally assumed that transforming the reading time values would not have a significant effect on the regression outcome.

Alternatively, the regression model can be specified to assume a different distribution, which reflects the distribution of the data better. In the case of reading time data, the gamma distribution would be a good fit, see centre subfigures in Figures 3.1 to 3.3. However, running the regression models specifying the gamma distribution was not possible for technical reasons[4].

[4]A practical problem occurred when trying to run regressions for a gamma-distributed response variable using R: it seems like there is a bug in the lmer function of the lme4 package that occurs when specifying the Gamma family. The regression exits with the error ”mu[i] must be positive”. This error has been observed by other researchers for this case as well, and reported to developers, but it has not been fixed as of beginning of August 2010. Alternative implementations of mixed-effect modelling for gamma distributed data is the glm function, which however does not allow the use of random effects, and the GenStat package, which turns out to be too slow to be used with large data sets like the one of interest for the work reported here.



(a) Model plot for raw reading times.



(b) Model plot for log reading times.

Figure 3.21: Model inspection with raw vs. log reading times as the response variable.

3.2.3 Correlation of Explanatory Variables

The underlying mathematical assumptions of regression models include that the explanatory variables be independent (assumption 4). That is, they should not capture the same effect and hence explain the same part of the variance. However, by the nature of some of our explanatory variables, this is not the case. For example, word frequency, forward transitional probability, and lexical Surprisal all depend on the frequency of a word and therefore capture partly overlapping aspects. Similarly, frequent words are usually short, while infrequent words tend to be longer etc. Therefore, it is important to determine whether there is a statistically significant correlation between different predictors. Table 3.1 shows that there are indeed strong correlations between the related predictors.

	word length	freq	word no	prev freq	land pos	launch dist	forw trans	backw trans	lexic surpr	ulex surpr
w-freq	**-0.70**									
w-number	0.03	-0.03								
prev-freq	0.07	-0.07	0.00							
land pos	**0.51**	0.18	-0.00	-0.06						
launch dist	-0.03	0.01	0.00	-0.00	-0.00					
forw.trans	**-0.56**	**0.67**	-0.01	-0.04	0.13	0.01				
back.trans	**-0.56**	**0.67**	-0.01	-0.03	0.15	0.01	**0.67**			
lex surpris	**0.51**	**-0.61**	0.02	0.01	-0.13	-0.00	**-0.68**	**-0.54**		
ulex surpr	-0.04	0.05	-0.02	-0.17	0.02	-0.00	-0.10	0.04	**0.35**	
integ cost	0.21	-0.28	0.03	0.03	-0.06	0.00	-0.22	-0.28	0.18	-0.07

Table 3.1: Correlations (according to Pearson test) between explanatory variables in the data set. Values are highlighted for correlations larger than 0.3.

Large correlations between predictors can cause large correlations between the estimated fixed effects. Such collinearity between fixed effects can lead to unstable results, where a coefficient estimate jumps around, i.e. it has a positive value in one model, but a negative one in a very similar model with the collinear predictor removed. Furthermore, significance estimates can be inflated. This means that coefficients and significances cannot be trusted for predictors which have large correlations with other predictors.

Strategies for removing collinearity in the model include centring predictors and

residualizing predictors against the ones they are correlated with, or expressing predictors differently. For example, the correlation between landing position on the word and word length is so strong because the length of the word strongly limits the values that landing position can possibly take on. Alternatively, the landing position can be expressed with respect to the word length, for example as the ratio $\frac{\text{landing position}}{\text{word length}}$. Given what we know about the IOVP effect, it also makes sense to assume a non-linear relationship between landing position and reading time. In fact, squared landing position values lead to a much better model fit, and make sense theoretically, as they model the shape of the IOVP effect. Correlation between the squared relative word landing position and word length is reduced to -0.25.

Baayen et al. (2008) also recommends running a kappa test on the predictors. If there is too much collinearity, the matrix of predictors could become singular, which would mean that the parameter estimation would be impossible. The kappa test determines the condition number, which estimates the degree to which the matrix is singular, meaning that there exists a potentially harmful degree of collinearity between predictors. If we run the kappa test on the variables which according to Table 3.1 show substantial correlation (word length, frequency, lexical Surprisal, structural Surprisal, forward transitional probability and backward transitional probability), we find that the condition number comes out as $k = 15.18$. As a guideline, Baayen suggests that a condition number between 0 and 6 suggests no collinearity, around 15 suggests medium collinearity and a condition number above 30 indicates potentially harmful collinearity.

The correlations between our predictors are hence slightly too high, so we will explore whether we can reduce them by residualizing. We residualize by running a linear regression between the predictor we want to residualize and the predictors that it is correlated with. If we residualize word length against word frequency, forward transitional probabilities against frequency, backward transitional probabilities against both frequency and forward transitional probabilities, and lexical Surprisal against word frequency, forward transitional probabilities and structural Surprisal, correlations between the resulting residualized predictors are removed, see Table 3.2.

Residualization of predictors however changes the interpretation of coefficients in the regression model: the model is for example not estimating the effect of word length on reading times, but the part of the word length effect that is independent of word frequency.

Finally, it is also important to keep in mind that we do not necessarily have to re-

	residual word length	word frequency	residual forward trans prob	residual backward trans prob	residual lexical surprisal
word frequency	0.01				
resid forward tp	-0.16	-0.03			
resid backward tp	-0.11	-0.02	0.04		
resid lex surpris	0.11	0.05	-0.03	-0.11	
structural surprisal	0.00	0.05	-0.19	0.08	-0.00

Table 3.2: Correlations (according to Pearson test) between residualized explanatory variables.

move collinearity between variables which we are not interested in. As long as they are not correlated with the variables that we are interested in, they will not change the coefficient and significance estimates of the predictor of interest. If we are only interested in a subset of the predictors, a safe and conservative method is to first estimate a model including all variables that we are not interested in, and run a second model with the residuals from the first model as the response variable. This way, there are no possible correlation effects between explanatory variables in the first and second model.

3.2.4 Dealing with Repeated Measures

Repeated measures refer to situations where measurements are collected under the same conditions multiple times. For our data, each subject read the whole corpus, and thus provided many data points. It can therefore be expected that the measurements taken from the same subject are related in some way, thus violating the assumption that errors are independent. Indeed, we have shown in Section 3.1.1.4 that the length of fixation durations, saccade sizes etc. depend on the subject, and that some subjects show stronger effects of some characteristics of the words than others.

Lorch and Myers (1990) compare three ways of dealing with repeated measures data. The first method simply averages over subjects so that there is just one data point for each item. This is also done in the traditional quasi-F testing where regressions are both run on the aggregated subject data points and on aggregated item data points. Effects are then only accepted to be significant if significance is reached in both tests. Lorch and Myers (1990) argue that averaging over subjects is not good practice for

regressing reading times: When subjects are averaged out, the variability associated with subjects is eliminated, and subjects are in fact treated as a fixed effect. Also, parameters such as landing position and launch distance have been shown to significantly influence reading times but are not available when subjects are averaged out. When using this method, Type 1 errors[5] get inflated. However, the mathematical problem of a non-normally-distributed response variable is less relevant in this case, because the averaging of the reading times causes the data to be more similar to a normal distribution (cf. Figure 3.22, as opposed to the more skewed distribution in Figure 3.3). Therefore, the models based on this data could be argued to be mathematically more reliable. A disadvantage of averaging over subjects is that we lose some predictors specific to the actual reading of that text by a specific human.

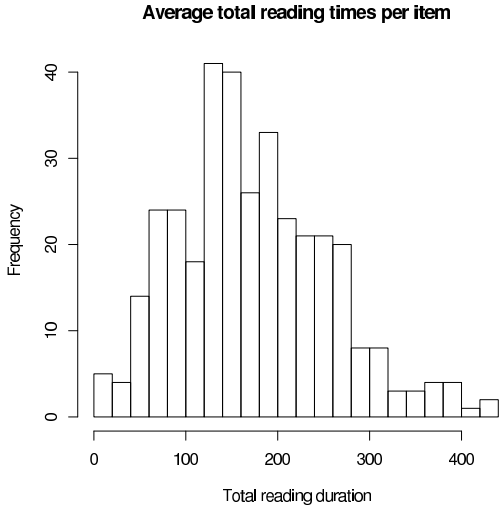


Figure 3.22: The distribution of total reading times when averaged across subjects.

Figure 3.23 shows other ways of model inspection for the regression with averaged subjects. The upper two plots show the distribution of residuals against fitted values. Ideally, there should be no pattern in the data, in particular, the data points should not lie within a triangular shape. The bottom left subfigure shows a quantile-quantile plot. If the points deviate from the straight line, this means that the data is skew (in the case of reading time data, it is skew to the right, and we therefore see a deviation from the linear line towards the top). Finally, the last subfigure in Figure 3.23 shows the leverage of the data points. From this plot one can read the influence of specific data points on the parameter estimates. The most influential points are those that have

[5] A Type 1 error is committed if we reject the null hypothesis when it is true, and Type 2 error is committed if we accept the null hypothesis when it is false.

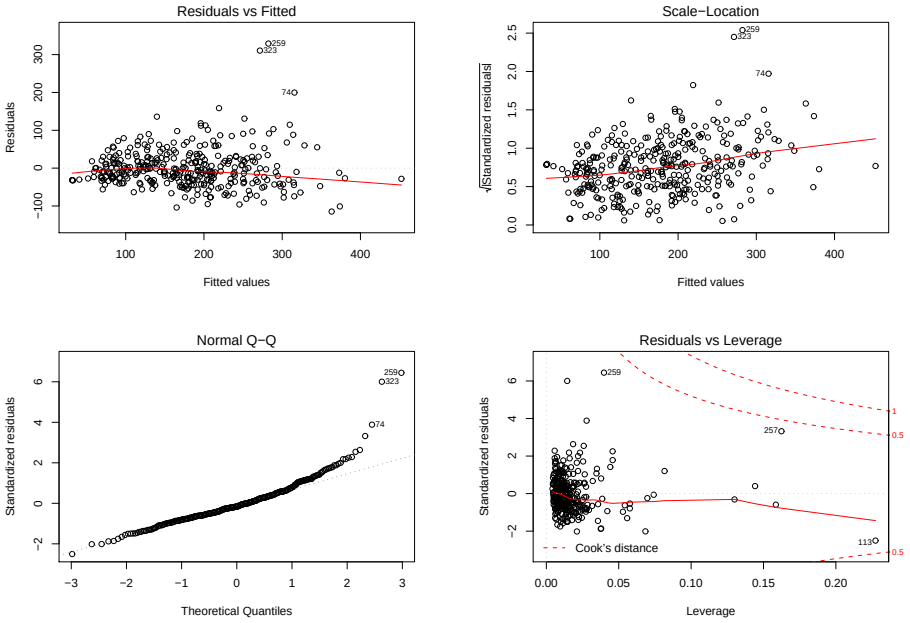


Figure 3.23: Error plot for regression of averaged total reading times.

a large Cook's distance. When a point is very influential, one should try leaving this point out of the model to see whether the estimates change substantially or not. We'll get back to outlier removal in the next section.

An alternative method is to run the regressions on the individual observations and to include the subject variable into the regression (effectively treating it as a fixed effect). But, as Lorch and Myers (1990) point out, this method also leads to inflated Type 1 error (although to a lesser extent than when averaging over subjects). Lorch and Myers (1990) therefore recommend to run separate regressions for each subject. However, Richter (2006) pointed out that there are some problems with the separate regressions method, because the data set is split up into subsets and is thus less reliable because of smaller data set size. Furthermore, variabilities of the separate regression estimates are not taken into account when running the t-test on the regression coefficients, which can also lead to biased results. Richter (2006) instead recommends to use hierarchical linear models.

In hierarchical mixed effects models, all data points are entered into the same regression equation, which has two (or more) layers. Participants were entered as a separate level from the items in the model, following Richter's (2006) recommendations for the treatment of reading time data (this is a generalisation of an approach initially proposed by Lorch and Myers (1990)). However, such a design might be slightly better

suited for situations where the nesting between random effects is inherent in the data, e.g. in some experiment where children would be nested under schools, which would in turn be nested under cities.

Mixed effect models with crossed subject and item random effects, as explained in Baayen et al. (2008) have recently become the new standard in the field. The difference between these models and hierarchical models is that hierarchical models assume a nesting between subjects and items effects, whereas Baayen's models "cross" subjects and items effects. In such mixed effects models with crossed random effects, separate intercepts and slopes are estimated for each item and each subject, if necessary (i.e. the model can of course be simplified if slopes do not help to explain the data). This means that the model can then determine whether e.g. there is a significant effect of word length on reading time which is common to all subjects, i.e. the random slopes give the model a way to adapt estimates to each subject, thus allowing for a situation where one subject's reading times are effected more strongly by word length than another subject's reading times. This work makes use of the lmer implementation of mixed effects models, which is part of the lme4 package (Baayen et al., 2008; Bates and Sarkar, 2007).

For our data, random effects under item were not estimated for several reasons. Firstly, estimating a separate intercept and slopes for each item (i.e. for each word in the corpus) is very likely to massively over-fit the data. Each word was only read by 10 subjects, and many of these data points are not present in the model due to track loss and skipping. Therefore, there simply aren't enough data points for sensibly estimating an intercept and slopes under item. Furthermore, the Dundee Corpus is different from typical psycholinguistic materials in that sentences were not constructed to test a specific effect. Therefore, the text in the Dundee Corpus corresponds more to a representative sample of the English language than typical experimental materials do. Finally, it is in practice not possible to include random intercepts and slopes for both subjects and items on our large data set because of memory restrictions, non-convergence of the model and extensively long run times. In particular, the model cannot estimate more than two slopes under subject if even just the intercept for item is included. We consider the slopes under subject to have a better theoretically motivated explanatory (and less over-fitting) effect on the data, and therefore include slopes under subject instead of random effects under item in the regression analyses reported in this thesis.

Compared to traditional quasi-F statistic analysis, mixed effects models are more

robust and conservative, which is why they should be preferred. However, for very small data sets, the quasi-F test can be more powerful, i.e. detect an effect more easily than a mixed effects model. This thesis analyses a very large data set, and thus focuses on the use of mixed effects models. Estimating random slopes for each subject makes model estimation very slow, and can lead to non-convergence or the model running out of memory if there is a large number of predictors and interactions, which are also included in the model as random effects. We found that it is too computationally expensive on the Dundee data set to include interactions as random slopes under subjects.

3.2.5 Outlier Removal

Outliers are points that are very atypical compared to the rest of the data. The problem with them is that they can have a strong influence on model estimations and lead to exaggerated or wrong estimates that don't reflect the patterns in the rest of the data set. We remove all data points that have too high leverage. Leverage is estimated as the difference in model estimations with and without each of the data points, called the difference in fits (dffits). When there's a large difference in estimations by just removing a single point, there is reason to consider that point as an outlier. Consider for example the plot in Figure 3.24 which plots the difference in fits for each point of a model for the Dundee Corpus. Baayen (2008) suggests to scale these differences in fit and then use a cutoff at 2.5 or 3 for removing points with high leverage, which is what we did for all regression models presented in this thesis.

3.2.6 Model Selection

Model selection refers to the process of choosing the model that best explains the data, i.e. choosing among the explanatory variables those that make a significant contribution to explaining the variance seen in the response variable.

A complete model would include all explanatory variables, all multi-way interactions between them[6] and all random slopes of explanatory variables under all random effects, including also random slopes for interactions.

One method to get to the model that best explains the data is to step-by-step remove

[6]Interactions between variables are when one variable modulates the effect of another explanatory variable. For example, we might find that beyond the main effects of word length and word frequency, frequency effects are stronger for short words than for long words.

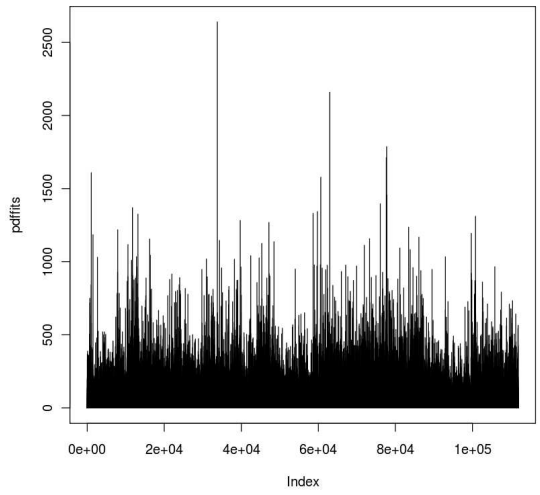


Figure 3.24: Leverage of all data points for a model of low-level predictors on the Dundee Corpus data.

from the complete model all predictors that do not significantly improve model fit, starting with multi-way interactions and ending with the main predictors. Here, it is important not to remove any main effect that is also a component in an interaction. Alternatively, one can use additive model building, starting with an empty model and step-by-step adding predictors if they improve model fit significantly.

Different versions of models can thereby be compared by calculating the log likelihood, degrees of freedom, Bayesian Information Criterion (BIC) and the Akaike Information Criterion (AIC) for each of the models. χ^2 is used to decide whether one model is significantly different from the other. The BIC and AIC scores combine both model fit and degrees of freedom of the model to calculate scores, with BIC penalising additional degrees of freedom more strongly than AIC. A model with more predictors will usually always fit the data better than one with less predictors, therefore, it has to be determined whether it is worthwhile to include the predictor, given the amount of the gain in model fit. So larger log likelihood is better than lower log likelihood scores, fewer degrees of freedom are better than many degrees of freedom, and lower AIC and BIC scores are better than higher ones. If two models are not significantly different, the one with fewer degrees of freedom is to be chosen, and if they are significantly different, the one with lower AIC (usually also coinciding with lower BIC) is chosen.

A mixed strategy for model selection was used, because it is impossible to estimate the complete model for our data set due to failing convergence and excessively long run times. For the models reported in this thesis, we started with a model including

all predictors as main effects, then added all possible binary interactions (but no other multi-way interactions) and all slopes under subject (we tested the model building also the other way around, adding slopes first and then interactions, but got identical results). Then any main factors that did not significantly contribute to the model were removed step-by-step (except, of course, for those main factors that are part of an significantly contributing interaction or slope). The final model contains only main effects, interactions and slopes that were significant contributors to the model.

3.2.7 Discussion

This section provided background about multiple linear regression models with fixed and random effects. We have discussed how to avoid violating assumptions of regression models by changing the distribution of the response variable and removing correlation between explanatory variables, and have motivated decisions of how these issues are dealt with in this thesis. Automatic methods for removing outliers from the analysis have also been motivated and discussed. Furthermore, we have given an overview of how to deal with repeated measures and concluded that a model with a by-subject random intercept and by-subject slopes are the best solution for our data.

We would like to take the last point up again for reflection. In the sections showing the distributions of fixation durations in the Dundee Corpus, and the discussion about the assumed normal distribution of the data, we have concluded that skipped words should be excluded from the data. This does however mean that short and frequent words, which are often skipped, are more difficult from the point of view of the model than they would be if skipping was factored in. It seems worthwhile bearing in mind that the difficulty of a word is also reflected in the skipping rate. As we have said, it is not possible to include the raw skipped values in the regression, and averaging across subjects, which would take care of the problem of combining fixation durations and skipped words, was ruled out based on argumentation in (Lorch and Myers, 1990).

There is no obvious perfect solution to this problem. In fact, the problem seems to lie at a deeper level: we should not try to directly fit difficulty estimates to reading times. Instead, there is an intermediary step which which we are missing: a model which translates processing difficulty into reading times, accounting for skipping of short frequent words even if they are unexpected, and spill-over effects on following words. Explanatory power of the explanatory variables would most likely be improved a lot this way. However, such a model is outside the scope of this PhD thesis.

3.3 Conclusions

This chapter has discussed the data set used for the experiments in this PhD, the Dundee Corpus, and the method for analysing it, linear mixed-effects models. We were able to show that the reading characteristics observed in the Dundee Corpus data are in line with previous findings. The data needs some more cleaning up than would be the case in a typical lab experiment due to the materials not being controlled and containing e.g. numbers, abbreviations and special characters.

The regression models reported in Chapters 4, 5 and 9.2 follow practices concerning residualization of explanatory variables, repeated measures treatment, outlier removal and model selection as discussed in the second part of this chapter. While we ran models on both log-transformed and raw reading times for all regression analyses conducted in this work, this thesis will usually report models on raw reading times, as results were equivalent, but raw reading times are easier to interpret.

Chapter 4

Case Study: Processing Difficulty in Naturally Occurring Relative Clauses

The goal in this chapter is to provide a proof of concept for using the Dundee corpus as a resource for evaluating theories for syntactic processing. It focuses on a very specific and relatively frequent structure, which has been investigated and discussed extensively in the literature, (e.g., King and Just, 1991): relative clauses. Being able to find the well-established results in the corpus is a good indication that it is possible to use the Dundee corpus as a complementary resource for testing theories, in addition to experimental test suites.

Early results of the work reported in this chapter were presented at CUNY 2007 and published at CogSci (Demberg and Keller, 2007).

4.1 Empirical Findings in Relative Clause Processing

Experimental results show that English subject relative clauses (SRCs) as in (1-a) are easier to process than object relative clauses (ORCs) as in (1-b). Experimentally, this difficulty is evidenced by the fact that reading times on region R1 in the SRC are lower than reading times for the corresponding region R2 in the ORC (King and Just, 1991), see Figure 4.1 for the original experimental results, obtained via self-paced reading. The SRC / ORC effect has also been found in a range of other studies to be a reliable effect in English (Gordon et al., 2001; Traxler et al., 2002) as well as other languages (Mak et al., 2002; Friederici et al., 1998). In recent work, Staub (2010) has shown that object relative clauses cause larger difficulty than subject relative clauses both on the embedded verb region (*attacked* in sentences from Example (1)) and on the NP

region in the relative clause (*the senator*). In this experiment, we however focus on the embedded verb region only.

(1) a. The reporter who [attacked]$_{R1}$ the senator admitted the error.
b. The reporter who the senator [attacked]$_{R2}$ admitted the error.

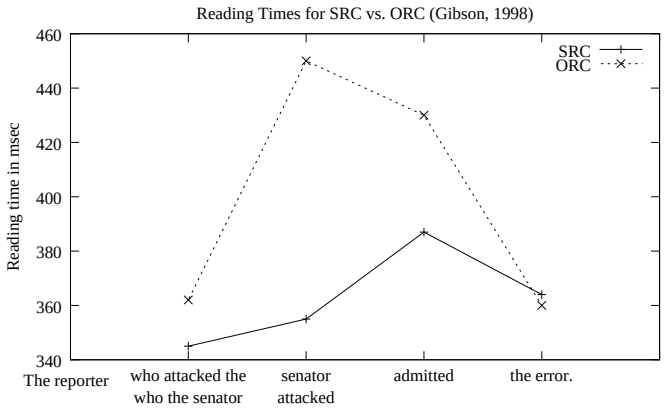


Figure 4.1: Results from the relative clause self-paced reading time experiment by King and Just (1991).

The size of the effect has been shown to also depend on factors related to the ORC noun phrase, e.g. animacy, semantic similarity to other entities in the context and topicality (Gennari and MacDonald, 2008; Gordon et al., 2004; Traxler et al., 2005; Reali and Christiansen, 2007).

The difference in processing difficulty on the embedded verb region in subject vs. object relative clauses cannot be explained by lexical factors (as the words in the two conditions are exactly the same) or higher syntactic ambiguity in the ORC condition (in fact, there is less ambiguity at the ORC embedded verb than at the SRC embedded verb). Findings such as these are explained by processing theories that capture the complexity involved in computing the syntactic dependencies between the words in a sentence. The most prominent such theory is Dependency Locality Theory (DLT), proposed by Gibson (1998, 2000) and explained in more detail in Section 2.2.2. DLT not only captures the SRC / ORC asymmetry while taking into account a notion of topicality (by counting discourse referents but not personal pronouns for calculating integration cost), but also accounts for a wide range of other complexity results, including processing overload phenomena such as centre embedding and cross-serial dependencies.

While DLT has been validated against a large range of experimental results, it has not been shown previous to this work that it can also successfully model complexity phenomena in naturally occurring text. Here we aim to test DLT's predictions on naturally occurring subject and object relative clauses rather than for isolated example sentences manually constructed by psycholinguists.

4.2 Experiment on RC Embedded Verb

4.2.1 Materials

For our data analysis, we used relative clauses from the Dundee Corpus (Kennedy and Pynte, 2005), see Section 3.1 for a detailed presentation and discussion of the corpus. We extracted all relative clauses headed by *who*, *whom*, *which*, *where*, *that*, and by PPs such as *for which*, and manually checked all sentences for whether they were indeed instances of relative clauses. We ended up with 502 relative clauses which we manually annotated for the position of the relative clause verbal region and the integration cost incurred at the RC verb. In relative clauses with auxiliaries or modals, we extracted the main verb of the relative clause, because this is where integration cost occurs according to DLT. In the case of predicative constructions, we extracted the inflected form of the predicative verb *be*[1]. The data contains about 25% object relative clauses and 75% subject relative clauses.

Reading times were computed for the different measures (first fixation time, first pass time, total reading time), as well as the total number of fixations in regions R1 and R2 for each item and subject. Linear mixed effects models using the reading measures as a dependent variable included only data points with fixation duration > 0 in the model, due to the reasons discussed in Section 3.1.1. There were 3046 data points for the total reading time analysis, 2608 data points for first fixation duration and first pass duration and 4056 data points in the regression estimating number of fixations.

4.2.2 Regression Procedure

The predictor we are most interested in for the study of processing difficulty in relative clauses is DLT integration cost (for a definition, see Section 2.2.2). However, as discussed in Section 2.1.2, it is well-known that reading times in eye-tracking data are

[1]Regression results turned out to be equivalent with and without these predicative items. These forms were often cliticised (*who'll, he's*) and were therefore unlikely to receive any fixation).

influenced not only by high-level, syntactic variables but also by a number of low-level variables that have to do with the physiology of reading, and lower level linguistic processing effects such as lexical access etc. In this study, we included the predictors listed in Table 4.1. All of these predictors were centred (i.e. the mean for a factor was subtracted from each value) to reduce collinearity.

predictor	value range	description
INTEGRATIONCOST	-1.44 – 5.56	manually annotated integration cost
RELATIVECLAUSETYPE	SRC, ORC	
RELATIVEPRONOUN	*that*, *who(m)*, *which*, *where*, *PREP which*	relativizer of the relative clause; *PREP which* summarizes prepositions followed by *which*, such as *of which*
RELATIVIZERTYPE	WHNP, WHPP, WHADVP	alternative coding for RELATIVEPRONOUN that groups *that*, *which*, *who(m)* in WHNP
BIGRAMPROBABILITY	-3.3 – 2.74	logarithmic; estimated from the BNC
LEXICALSURPRISAL	-7.4 – 13.6	estimated from Roark (2001a)
STRUCTURALSURPRISAL	-2.1 – 9.3	estimated from Roark (2001a)
WORDLENGTH	-3.29 – 6.7	in characters
SENTENCEPOSITION	-15.6 – 45.4	the position of the word within the sentence (counted in words)
WORDFREQUENCY	-3.6 – 1.9	logarithmic, estimated from the BNC
PREVFIX	yes, no	the flag marking whether the previous word had been fixated
FREQOFPREV	-5.3 – 1.1	the frequency of the previous word to model spill-over effects
LAUNCHDISTANCE	-2.0 – 8.7	difference from current to previous landing position in letters
LANDINGPOSITION	0.00003 – 0.5	squared word landing position relative to word length to model the IOVP effect

Table 4.1: Predictors for the linear mixed effects models for reading times on the RC verb, and their value ranges after centring. Frequency estimates are per million words.

For each of the continuous dependent variables (total reading time, first fixation duration, first pass duration), we ran separate mixed effect linear regressions that included the independent variables, interactions and random slopes under subject, as described in Section 3.2. Final models were determined using the model selection techniques explained in Section 3.2.6.

4.3 Results

4.3.1 Total Reading Time

The distribution of total reading times in the embedded verb region turned out to be very skew, and significantly different from the normal distribution, while log transformed reading times were not significantly different from a normal distribution. We therefore ran all models on the log transformed reading times.

The final model for log transformed total reading times includes INTEGRATIONCOST, FREQUENCY, WORDLENGTH, FREQOFPREV, LAUNCHDISTANCE, LANDINGPOSITION as main fixed effect, an interaction between word length and landing position WORDLENGTH:LANDINGPOSITION and a random intercept and random slope for FREQUENCY under SUBJECT.

Random slopes under item did not improve the model. As pointed out before, this is potentially due to over-fitting. According to the Bayesian information criterion (BIC), models including random slopes under item were consistently rated worse than models not including random slopes under item. Furthermore, adding two or more random slopes under item at the same time would lead to non-convergence of the model.

We removed outliers from the model by identifying points with large leverage, as explained in Section 3.2.5. The results for fixed effects of the final model are shown in Table 4.2. Centring the predictors lead to a big reduction in correlation for some variables – compare correlations between fixed effects before and after centring in Table 4.3. To make sure that the remaining correlations were not influencing our factor of interest, INTEGRATIONCOST, we fitted a model including all other factors and then ran a regression of INTEGRATIONCOST on the residuals. Regression coefficient and significance level on residuals were exactly identical with the main model ones, so we conclude that the remaining level of collinearity does not affect model interpretation regarding INTEGRATIONCOST.

The main effect of relative clause type did not significantly improve the model, and was thus removed from the final model. However, integration cost, which has previously been shown to correctly predict the difficulty incurred in relative clauses, and can be regarded as a more fine-grained measure than simple SRC / ORC flags, significantly improves model fit and reaches significance as a positive predictor for total reading times: the model adjusts its estimation of total reading time upwards for items with higher integration costs. Random slopes for integration costs under subject and item did not improve model fit and therefore were not included in the final model.

Predictor	Coefficient	Significance
(INTERCEPT)	5.43	***
INTEGRATIONCOST	0.01	*
WORDLENGTH	0.03	***
WORDFREQUENCY	-0.07	***
FREQOFPREV	0.01	
LANDINGPOSITION	-0.19	**
LAUNCHDISTANCE	-0.01	***
WORDLENGTH:LANDINGPOSITION	-0.12	***

$.p < 0.1$, $*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 4.2: Log-transformed total reading times for the embedded verb in relative clauses – coefficients and their significance levels for a reduced model based on a complete model including main effects, two-way interactions and slopes for both items and subjects.

	(Intr)	IntegCost	Length	Freq	LDist	Surpr	LandPos
raw predictors:							
INTEGCOST	-0.234						
LENGTH	-0.437	-0.081					
FREQ	-0.712	-0.032	0.307				
FREQOFPREV	-0.504	0.304	-0.103	-0.040			
LANDPOS	-0.141	0.057	0.452	-0.069	0.004		
LAUNCHDIST	0.143	-0.011	0.077	0.031	0.019	0.002	
LENGTH:LANDPOS	0.092	-0.062	-0.421	0.076	0.021	-0.938	0.012
centred predictors:							
INTEGCOST	-0.004						
LENGTH	-0.069	-0.079					
FREQ	-0.259	-0.030	0.307				
FREQOFPREV	-0.032	0.303	-0.102	-0.040			
LANDPOS	-0.235	-0.018	0.058	0.024	0.071		
LAUNCHDIST	-0.008	-0.011	0.079	0.031	0.020	0.031	
LENGTH:LANDPOS	0.007	-0.061	-0.420	0.076	0.02-	-0.246	0.013

Table 4.3: Correlations between fixed effects of the fitted model (i.e. these are not the correlations between explanatory variables) for raw and centred versions of predictors.

Integration costs are significantly higher for ORCs (mean $= 2.39$) in our data than for SRCs (mean$= 1.18$; $t = 19$, $p < .0001$). From this, we conclude that we can find the SRC / ORC asymmetry effect in corpus data, when measuring it with the more fine-grained integration cost measure rather than just the SRC / ORC flag.

The results of a regression model using non-transformed reading times are very similar to the results from the model with log-transformed reading times, but some significance values are slightly different. As the non-transformed reading time model is easier to interpret, the next section will explain what the model estimations mean using the non-transformed model, as shown in Table 4.4.

Predictor	Coefficient	Significance
(INTERCEPT)	262.54	***
INTEGRATIONCOST	4.44	.
NPTYPE=PRON	-11.51	*
WORDLENGTH	8.53	***
WORDFREQUENCY	-19.57	***
FREQOFPREV	2.65	
LANDINGPOSITION	-63.60	**
LAUNCHDISTANCE	-2.78	***
WORDLENGTH:LANDINGPOSITION	-34.14	***

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 4.4: Total reading times (not log transformed) for the embedded verb in relative clauses.

During model estimation, the estimation algorithm of the model tries to weigh each predictor such that the best fit with the dependent variable (here, the total reading times) is obtained when adding up all weighed predictors. Let us now go through how the reading time for a word is estimated given the values of the predictors and the model estimates. The intercept is 262.54 (see Table 4.4), which means that a verb has a predicted base reading time of 262.54 ms. One would then add 4.44 times the (centred) integration cost at the verb in ms, add 8.53 times the (centred) word length in ms and subtract 19.57 ms for each log frequency unit of the word. The launch distance is the distance in letters between the current fixation and the previous fixation. Hence, when the eye moves to the right, the launch distance has a negative value. The negative coefficient for launch distance thus means that a longer launch distance leads

to longer reading times (2.78 ms for each letter-distance). The squared relative landing position value (fixation positions at the edge of a word have a larger value than fixation positions toward the middle of the word) is multiplied by 63.60 and subtracted from reading times. The large value for landing position is caused by the fact that landing position is calculated relative to word length, then centred and then squared, so values only range from 0 to 0.5, while most other factors have a larger range – for an overview of the distribution of the explanatory variables, see Chapter 3.1. Finally, the interaction between word length and landing position is added to the reading time estimations as the product of centred word length, squared relative landing position and the coefficient -34.14. This means that the IOVP effect is stronger for long words. Neither lexical nor structural Surprisal came out as a significant predictor on this data set.

The model also contains an intercept and a slope for frequency under subjects. The random intercept under subject means that the model estimates a different base reading time for each subject – some people are faster readers who can be expected to spend less time on a word on average than others. Fitting a frequency random effect under subject means that for each subject, we allow for a slightly different effect of word frequency, which means that the model estimates in how far each subject is affected by differences in word frequency. Rare words might slow down some readers more than others.

4.3.2 Early measures

The distribution of early measures is not as skew as the distribution of total reading times, we are therefore going to report models that use raw reading times as the response variable.

The model for first fixation duration was determined in the same way as described for the total reading time model. The final model contains the predictors WORDLENGTH, PREVFIX, WORDFREQUENCY, SENTENCEPOSITION, LANDINGPOSITION, LAUNCHDISTANCE and an interaction between, again, the length of a word and the relative landing position of the first fixation on it. Only WORDLENGTH turned out to significantly improve the model when added as a random slope under item. Again, centred versions of all explanatory variables were used to remove collinearity, and data points with atypically high leverage were removed from the final model. The resulting model is shown in Table 4.5.

In first pass times, there was no significant effect for frequency of the last word, but

Predictor	Coefficient	Significance
(INTERCEPT)	247.75	***
INTEGRATIONCOST	9.00	*
WORDLENGTH	9.63	***
WORDFREQUENCY	-7.09	.
PREVFIX	-41.88	***
LAUNCH DISTANCE	-1.18	
SENTENCEPOSITION	-1.13	*

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 4.5: Final model of first pass durations for the embedded verb in relative clauses – coefficients and their significance levels.

instead an effect of whether the last word had been fixated (PREVFIX). Furthermore, we did not find an effect of any interactions, or landing position in first pass times. INTEGRATIONCOST came out as a significant positive predictor of reading times in first pass reading times, confirming the effect seen on total reading times. Results for log transformed reading times as a response variable yielded the same results. There was no significant effect of either structural or lexical Surprisal.

On first fixation times, there is no significant effect of integration cost, but it remains in the model as a random effect under subject which significantly improves the model. The lack of a main effect of integration cost in first fixation times is not surprising, given that related work has usually only found a reliable effect of relative clause type on late measures such as total reading time, but not on first fixation time, which is a very early measure. Significant predictors for first fixation times included only low-level explanatory variables. Again, model fit was not improved by adding lexical or structural Surprisal as a predictor of first fixation times.

Regression Model for Number of Fixations In the above regression analyses, data points where no fixation had taken place within the definition of the dependent reading time variable were not included in the analyses. In a last analysis, I therefore tested whether relative clause type is predictive of how often the embedded verb region is fixated during reading. We therefore ran another model with dependent variable NUMBEROFFIXATIONS and the usual predictors, except those that are indirect indicators of the number of fixations, such as LAUNCHDISTANCE and LANDINGPOSITION.

The model resulting from model selection is not very interesting. Only word length

and frequency came out as reliable predictors for the number of fixations on a word (as expected, long words are more likely to be fixated than short ones, and frequent words are fixated less often than infrequent ones). In addition, skipping probabilities are almost identical for subject and object relative clauses: they amount to about 36% for first pass skipping (i.e., the word is skipped before a word to the right is fixated) and 25% for total skipping (i.e., the word is never fixated).

4.4 Discussion

As expected, a significant proportion of the data is explained by low-level factors such as word length, the frequency of a word and oculomotor-related effects such as fixation landing position and launch distance. We were not able to find any significant effect of lexical or structural Surprisal on the critical region. A possible explanation for this lack of effect is that Surprisal has been argued to make incorrect predictions for the embedded verb of English relative clauses, predicting that a verb in the ORC condition should be easier to read than a verb in the SRC condition. Given this expectation of Surprisal not being a good predictor for reading times in relative clauses, it is maybe not astonishing that no significant effect was found in this experiment.

DLT integration cost however has been argued to correctly model the SRC/ORC asymmetry, and we did find a significant effect of DLT integration cost in total reading times and first pass times. This effect cannot easily be explained away by e.g. spill-over effects, as predictors relating to the difficulty of the preceding word such as the frequency of the previous word and whether it had been fixated, were included in the models as explanatory variables, but did not cause the integration cost effect to go away. We did not find a statistically significant effect of integration cost in first fixation durations, which indicates that the effect must be driven also by refixations on the critical region.

The results presented here provide evidence for processing differences between subject and object relative clauses in naturally occurring sentences in context. While no main effect of the binary distinction SRC / ORC was found, this chapter has argued that the statistically significant positive integration cost effect is a more fine-grained measure of the processing difficulty differences in subject and object relative clauses. We showed that integration cost at the embedded verb of an object relative clause is significantly higher than at the embedded verb of a subject relative clause, and that higher integration costs lead to longer reading times, which in turn reflect larger pro-

cessing difficulty. The integration cost effect occurred on total reading times and first pass times, but not first fixation times, which is in line with previous results experimental results on SRC / ORC processing from the literature. Our finding that the more fine-grained measure of integration cost, which takes into accoun aspects of topicality, was a far better predictor of reading times than a simple SRC / ORC flag, seems also very compatible with recent findings of reduced object relative clause difficulty for object relative clauses in context, see Section 4.4.1.

Having replicated well-established experimental findings on the difference in processing difficulty between subject and object relative clauses in English, the Dundee Corpus seems to be a valid and valuable resource for testing theories of sentence processing in addition to traditional controlled experiments.

4.4.1 Related Work on Contextualised Relative Clause Processing

Since we first ran initial experiments on relative clauses in the Dundee Corpus, some other groups of researchers have also published work addressing the question of whether the traditional relative clause findings hold for contextualised text or is an artefact of the single-sentence presentation in experimental designs.

Mak et al. (2008) argued that the processing difficulty observed in object relative clauses is due to wrong topicality when the relative clauses are presented without a suitable context. They argue that in naturally occurring text, unlike experimental sentences, object relative clauses are chosen to fit the topic structure of the discourse, and that difficulty occurs when violating this topic structure, as is the case with object relative clauses presented without context in experiments. Mak et al. found that the processing of object relative clauses was greatly facilitated if they were licensed by the preceding discourse structure. This finding fits with the formulation of DLT integration costs in that integration costs are calculated in terms of how many new discourse elements intervened between an argument and its head. If the NP in an object relative clause was the discourse topic, the NP is usually a personal pronoun and thus does not cause any integration cost. Similarly, Reali and Christiansen (2007) showed that relative clauses in naturally occurring text often have a relative pronoun as the noun phrase, and that processing such frequently-occurring patterns of ORCs is easier than the object relative clauses used in psycholinguistic experiments, that typically have a full noun phrase. In his 2007 CUNY poster, Roland et al. (2007) argued similarly that the difference in processing difficulty between subject and object relative

clauses is due to experimental design and disappears when relative clauses are presented in context. These findings are seemingly in contrast to our publication at CogSci (Demberg and Keller, 2007), where we did find a significant ORC difficulty effect for relative clauses in context. Douglas Roland has since extracted relative clauses from the Dundee Corpus and argued that our findings are due to a single outlier (Roland, 2008) and failure to use a random slope for relative clause type under item (Roland, 2009). We were not able to replicate Roland's null-results by removing the outlier sentence from our data, presumably due to the fact that Roland had extracted a slightly different set of sentences from the corpus.

All challenges put forward by Roland have been addressed in the analyses reported in this chapter: we have checked all automatically extracted relative clauses by hand, to make sure they are all indeed relative clauses, and guarantee that critical regions are annotated correctly[2]. Furthermore, we have added prepositional and adverbial (*where*) relative clauses to increase the amount of data points available for the object relative clause case. Any data points with large leverage are removed from the data set as outliers, and we included slopes for random effects of subject where they improve model fit. Concerning random slopes under item however, we found that model fit and number of data points per item indicate that including random slopes under item would lead to problems of over-fitting the data. We find that even if we include it, such a slope neither improved model fit ($p \approx 0.99$), nor does it change the outcome of the integration cost main effect.

4.5 Conclusions

We were able to show that the integration cost component of Dependency Locality Theory ((DLT) Gibson, 1998, 2000), correctly predicts differences in processing complexity for subject and object relative clauses. The complexity effect was tested on the embedded verb of the relative clause and lead to elevated reading times in the ORC condition.

When an early version of this experiment was first published, this was the first time a theory of sentence processing had been tested on data from an eye-tracking corpus. Since, other researchers have started using the Dundee Corpus and the Potsdam Sentence Corpus (which is a collection of sentences rather than a corpus of naturally occurring text) for similar studies.

[2]Big thanks to Frank Keller and Roger Levy for helping me with this chore!

While this chapter has only dealt with one construction (relative clauses), we believe that our corpus-based approach constitutes an important new methodology for evaluating models of sentence processing. Such models were previously tested exclusively on data obtained for isolated, manually constructed sentences in controlled lab experiments. The validity of the models can be enhanced considerable if we are able to show that they scale up to model reading data from an eye-tracking corpus of naturally occurring text. This task is tackled in the next Chapter of this thesis, where DLT and Surprisal are tested on the complete data of the Dundee Corpus.

Chapter 5

Broad Coverage Evaluation of DLT and Surprisal

This chapter evaluates two previous theories of sentence processing, DLT and Surprisal, on the Dundee Corpus. Experiments on relative clauses in the last chapter showed that a well-known effect, the SRC/ORC asymmetry can be observed also on naturally occurring relative clauses from the Dundee corpus, and that DLT integration cost correctly predicts the data, while Surprisal did not. Here, we want to extend the comparative evaluation of integration cost and Surprisal to the full range of constructions in the Dundee Corpus. The main question we want to answer is whether these two prominent theories of sentence processing, which have been shown to successfully model a range psycholinguistic effects, also predict processing difficulty on naturally occurring text.

This chapter reports four experiments. The first experiment evaluates integration cost on the whole data set, and the second experiment analyses integration cost in more detail on the types of words where the bulk of the predictions are made: nouns and verbs. The third experiment evaluates two versions of Surprisal, lexical Surprisal and structural Surprisal on the whole data set. The last experiment presents a comparative analysis of Surprisal and DLT integration cost as predictors added to the same baseline model. In the last part of the chapter, implications from these experiments are discussed.

The material described in the chapter was presented at AMLaP 2007 and published in Cognition (Demberg and Keller, 2008a).

5.1 Motivation

As described in Section 2.2, a number of different theories of syntactic processing complexity exist. However, this study will focus on DLT and Surprisal, as these two approaches are maximally different from each other. In particular, they make complementary assumptions about the source of processing complexity. DLT's integration cost captures the cost incurred when a head has to be integrated with the dependents that precede it, see Section 2.2.2. Surprisal, on the other hand, accounts for the cost that results when the current word is not predicted by the preceding context, see Section 2.2.4. Therefore, integration cost can be regarded as a backward looking cost (past material has to be held in memory and integrated), while Surprisal is a forward-looking cost (syntactic predictions have to be discarded if they are no longer compatible with the current word). This observation leads to a general empirical prediction, viz., that integration cost and Surprisal should be uncorrelated, and should account for complementary aspects of reading time data. The present study will test this prediction.

While DLT and Surprisal have been evaluated against a range of experimental results, so far no *broad coverage* evaluation of theories of syntactic processing complexity has been carried out. Existing studies rely on lab experiments, which have the advantage of giving the experimenter full control over the experimental setup and the materials, and are of established reliability and validity. However, this approach also has its drawbacks. It typically involves the presentation of artificially constructed sentences containing a narrow range of syntactic structures. Also, the same structures occur many times in a given experiment, raising the possibility of habituation effects or the development of strategies in participants. The sentences to be tested are often presented in isolation, i.e., without an appropriate textual context, potentially leading to behaviour that is different from normal reading. Finally, only a small number of items can be tested in the typical psycholinguistic experiment. DLT and Surprisal effects have successfully been obtained in such a experimental settings, but these methodological limitations leave open the possibility that the effects are rare or absent when arbitrary words in large amounts of naturalistic, contextualised text are considered.

The aim of the study presented in this chapter is to address this problem and provide a broad coverage evaluation of DLT and Surprisal on the Dundee Corpus, a large collection of newspaper text for which the eye-movement record of 10 participants is available, see Chapter 3.1. From this corpus, a range of eye-tracking measures can be computed, which can then be evaluated against the predictions of theories of syn-

tactic processing complexity. Such broad coverage studies yield results that hold for naturalistic, contextualised text, rather than for isolated example sentences artificially constructed by psycholinguists. Chapter 4 showed how individual phenomena, such as the subject/object relative clause asymmetry can be detected in naturally occurring text. The aim of the broad-coverage evaluation presented in this chapter is to show that corpus studies can also be used to systematically test theories of syntactic processing complexity. Such studies provide a source of evidence that corroborates experimental results, but also yields new theoretical insights, as it makes it possible to evaluate multiple theoretical predictors against each other on a large, standardised data set.

5.2 Predictors of Processing Difficulty

In this study, we are primarily interested in how far two existing theories, DLT (see Section 2.2.2) and Surprisal (see Section 2.2.4), can account for the reading times in broad-coverage text. We implemented the integration cost component of DLT, based on the dependency parses from the MINIPAR parser (Lin, 1998), and calculated Surprisal based on the Roark parser (Roark, 2001a). We then automatically calculated difficulty predictions by the two theories for each word in the Dundee Corpus.

Reading times in eye-tracking data are influenced not only by high-level, syntactic variables but also by a number of low-level variables, both linguistic ones and oculomotor ones, as discussed in Section 2.1.2. Together with variation between readers, these low-level variables account for a sizable proportion of the variance in the eye-movement record. It has also been shown that information about the sequential context of a word can influence reading times. In particular, McDonald and Shillcock (2003b) present data extracted from an eye-tracking corpus (a smaller corpus than the Dundee Corpus used here) that show that forward and backward transitional probabilities are predictive of first fixation and first pass durations: the higher the transitional probability, the shorter the fixation time. For a more detailed explanation of transitional probabilities, please refer to Section 2.2.5.

In this study, we are interested primarily in syntactic processing effects such as the ones captured by DLT integration cost and Surprisal. We therefore need to make sure that these metrics account for variance in the eye-movement record that is not captured by the low-level linguistic and oculomotor variables. Technically, this can be achieved by running mixed effects models which include both the low-level and the high-level variables as predictors, as well as partitioning out subject variance, see Section 5.3.1.2.

5.3 Experiment 1: Integration Cost

The aim of this experiment is to provide a broad-coverage test of Gibson's DLT by investigating whether integration cost is a significant predictor of eye-tracking measures obtained on a corpus of naturally occurring, contextualised text.

5.3.1 Method

5.3.1.1 Data

For our data analysis, we used the English portion of the Dundee Corpus (Kennedy and Pynte, 2005), whose characteristics have been described in Chapter 3.1. Before carrying out our analyses, we excluded all cases in which the word was the first or last one of the line, and also all cases where the word was followed by a any kind of punctuation. This eliminates wrap-up effects that occur at line breaks or at the end of sentences. Furthermore, we excluded all words that were in a region of four or more adjacent words that had not been fixated, since such regions were either not read by the participant or subject to data loss due to tracking errors, and all strings including digits, special symbols or several upper case letters. This left us with $\approx$ 383k data points, of which about 240k were fixated at least once, and of which about 200k were fixated during first-pass reading.

5.3.1.2 Statistical Analysis

The statistical analyses in this chapter were carried out using linear mixed effects models (Pinheiro and Bates, 2000). These models can be thought of as a generalisation of linear regression that allows the inclusion of random factors (such as participants or items) as well as fixed factors. The fixed factors can be discrete (such as whether the previous word was fixated) or continuous (such as word frequency). The models reported here include a random intercept and slopes under SUBJECT, as suggested in Baayen et al. (2008), see Section 3.2. However, intercept and slopes for ITEM were not included in the models, as there are too big risks of over-fitting the model: there are no repeated items in the sense of a psycholinguistic experiment. Each item (i.e. word in the corpus) was only read by 10 subjects, and in many cases less than 10 data points are available to the regression model due to track loss and skipping. Hence, an analysis including random effects by item does not seem applicable in this setting; refer back to Section 3.2.4 for a more complete discussion.

A separate mixed effects model was computed for each of the three dependent variables (first fixation duration, first pass duration, and total reading time), including low-level explanatory variables as well as transitional probabilities and integration cost as predictors. Minimal models were obtained following the model reduction methods outlined in Section 3.2.6.

In the remainder of the chapter, we will give the coefficients and significance levels for those predictors that remain in the minimal model. All of these coefficients are statistically significant, with the possible exception of main effects, which are only removed from the minimal model if there is no significant interaction that depends on them.

5.3.1.3 Implementation

Non-syntactic Predictors The non-syntactic predictors used were word length in characters (WORDLENGTH), word position in the sentence (SENTENCEPOSITION), whether the previous word was fixated (PREVIOUSWORDFIXATED), the distance between the previous fixation and the current fixation (LAUNCHDISTANCE), and the square of the position of the character on which the eye lands in the word, relative to word length (LANDINGPOSITION). The square of the centred relative word landing position was used to model the IOVP effect, see Section 3.1.2.2. These values can be read off directly from the Dundee Corpus (with the exception of SENTENCEPOSITION which we calculated after automatically determining sentence boundaries for the Dundee Corpus text). The predictors logarithmic word frequency (WORDFREQUENCY), logarithmic word frequency of the previous word (PREVIOUSWORDFREQUENCY), forward transitional probability (FORWARDTRANSITIONALPROBABILITY), and backward transitional probability (BACKWARDTRANSITIONALPROBABILITY) need to be estimated from a training corpus. We used the British National Corpus (BNC) (Burnard, 1995) and estimated unigram and bigram probabilities using the CMU-Cambridge Language Modelling Toolkit (Clarkson and Rosenfeld, 1997). For the bigram model, many of the bigrams from the Dundee Corpus were not observed in the BNC training data. To avoid having to assign a bigram zero probability just because it did not occur in the training data, we smoothed the bigram probabilities, i.e., some of the probability mass of the seen events was redistributed to unseen events. We used the Witten-Bell smoothing method (Witten and Bell, 1991), which is predefined in the CMU Toolkit. For a more detailed discussion of estimating word frequencies, see Section 3.1.2.4.

Integration Cost It is not feasible to manually compute values for the predictor integration cost (INTEGRATIONCOST) for the whole Dundee Corpus, given its size. We therefore relied on automatic methods which can handle a large amount of data (but are potentially error-prone). We parsed the corpus with an automatic parser and implemented a function that uses these parses to assign integration cost values to the words in the corpus. The parser used was MINIPAR (Lin, 1998), a broad-coverage dependency parser for English. MINIPAR is efficient and has good accuracy: an evaluation with the SUSANNE corpus (Sampson, 1995) shows that it achieves about 89% precision and 79% recall on dependencies (Lin, 1998). A dependency parser was chosen because the dependency relationships that it returns are exactly what we need to calculate integration costs (see Figure 2.3 for an example).

A practical issue was that tokenization in the Dundee corpus is often different from the tokenization used by the parsers. Therefore, it is necessary to realign the parsed text with the Dundee corpus segmentation. If a word in the Dundee corpus corresponds to multiple words in the parsed version, we have to combine the theories' predictions for those two words into a single prediction for that token, or split up the Dundee token into two bits. We here decided to combine the predictions for two different words into a single value and use the Dundee corpus tokenization. Integration costs of two words that were just one token in the Dundee Corpus were combined by simple addition, because the relevant quantity is the combined integration cost of the two components, which means that e.g. averaging would not be an appropriate measure.

In our implementation, integration costs are composed of the cost of (a) constructing a discourse referent and (b) the number of discourse referents that occur between a head and its dependent, excluding the head and the dependent themselves. This requires discourse referents to be identified in the corpus; we used the approximation that all words that have a nominal or verbal part of speech are discourse referents. Using part of speech tags assigned by the parser also allows us to differentiate between auxiliaries, modals and full verbs, and to automatically identify nouns that are parts of compound nouns. It is important to note that two versions of integration cost exist in the literature: one based on Gibson's (2000) DLT, and the earlier version based on Gibson's (1998) syntactic prediction locality theory, a predecessor of DLT. The difference between the two versions concerns nouns; here, we assume the Gibson (2000) version of integration cost (though we conducted some experiments with the 1998 version, see Section 5.4.3). DLT has later been extended and revised to provide a more extensive account of noun phrases (e.g., Warren and Gibson (2002)), but this revised

version of DLT has not been formalised, and thus would be hard to implement without making additional assumptions.

We evaluated our integration cost implementation using a short text that had been hand-annotated with integration cost values by Asaf Bachrach. The text was also used in e.g. (Roark et al., 2009). This evaluation gives us an estimate of how well our automatic annotation tool performs. We found that the integration cost values assigned automatically to the 764 words in the evaluation text were correct 83% of the time. Further analysis revealed that the automatically assigned integration cost values were significantly correlated with the manually assigned ones (Pearson's $r = 0.697$, $p < 0.001$). This result needs to be regarded as a lower bound. Unlike the Dundee Corpus, the evaluation text was not a newspaper text. Rather, it was a manually constructed story created in order to contain sentences with high integration cost. The sentences in the evaluation text are often long and complicated, and therefore hard to analyse with our automatic tool. Mean integration cost in the evaluation text was 0.7, while in the Dundee Corpus it was 0.55.

5.3.2 Results

In Experiments 1 and 2, we will only consider results for first pass durations in detail. The results for first fixation durations and total times are broadly similar, and will only be discussed briefly. We will return to this in Experiment 3, which provides a comparison of the results for the three eye-tracking measures for a model that contains all the predictors used in this chapter (see Section 5.5.3).

Table 5.1 shows the coefficients and significance levels obtained when running linear mixed effects models on first pass durations extracted from the Dundee Corpus. The models includes all the non-syntactic predictors and integration cost, and were computed over all words in the corpus, as well as significant interactions. Collinearity analysis by inspection of correlations between fixed effects after fitting the model showed that bigram forward transitional probability was correlated with unigram word frequency, and that backward transitional probability was correlated with forward transitional probability and unigram frequency. We therefore residualized forward transitional probability by calculating the residuals of forward transitional probability in a regression against word frequency, and residualized backward transitional probability by regressing against both word frequency and forward transitional probabilities. There is also some collinearity between word length and word frequency, and between

the frequency of the last word and the flag for fixating it. As these variables are not of direct interest in our study, and don't strongly correlate with any of our predictors of interest, we did not attempt to remove this collinearity, as it shouldn't affect our conclusions. For a full correlation table between fixed effects for the model in Table 5.1, see Table 5.2. We also removed outliers, as discussed in Section 3.2.5. We ran the model both with just a random intercept under subjects and with the full range of random slopes (all main predictors) under subject. The results were equivalent in terms of coefficients for the main effects. Significance values tended to be a bit lower for the model including random slopes under subject, but all remained strongly significant at $p < 0.001$ with the exception of residual forward transitional probability which did not reach significance any more. We here report the model without random slopes, because the model with random slopes showed high collinearities between predictors, even if these predictors had been residualized before or are not significantly correlated at all. Furthermore, there were very strong correlations between a number of fixed effects ($r < 0.8$). Confidence in the results of the simpler model rests on the fact that main effects remained unchanged with respect to the model that includes random slopes. Collinearity can lead to inflation of coefficients and significance values, and to unstable results with coefficients jumping between e.g. positive and negative values. Therefore, a model with high collinearity cannot be interpreted reliably. On the other hand, collinearity must not bother us as long as it does not affect the predictors we're interested in. Therefore, we conducted a second analysis, which can be regarded as very conservative: A model including slopes under subject was first fitted for all predictors except the one we are interested in. Then, a model with just the predictor of interest, i.e. just INTEGRATIONCOST and a slope of INTEGRATIONCOST under subject was fitted on the residuals of the first model. This way, it is guaranteed that the fixed effect of the predictor of interest is not correlated with any of the other predictors. Results of this model, which confirmed the significant negative effect of integration cost, are reported in Section 5.6.

Our findings confirm many effects also found by other researchers. Table 5.1 shows an intercept of approximately 243 ms. This can be regarded as the base reading time of an average word, to which the value for each predictor multiplied by the coefficient for that predictor is added to obtain the predicted reading time for that word. For example, the coefficient of WORDLENGTH is approximately 8 ms. As the predictor was centred, this means that for a word which is one letter longer than average, an additional 8ms is added to the estimate. The fact that the coefficient of WORDLENGTH is positive

means that longer words have longer reading times, a basic finding in the reading literature. We furthermore observed a negative coefficient for logarithmic word frequency (WORDFREQUENCY), which means that more frequent words are read faster than less frequent words.

The variable PREVIOUSWORDFREQUENCY was included in the analysis to account for possible spill-over effects, where a previous difficult word causes longer reading times on the current word. Indeed, we found that the log-frequency of the previous word was a significant negative predictor of reading time: if the previous word is rare, reading times are expected to be longer on the current word. We also find that the presence of a fixation on the previous word (PREVIOUSWORDFIXATED) reduces reading time on the current word by 35 ms, i.e., fixation time is longer when the previous word was skipped. There is also an effect of squared relative landing position (LANDINGPOSITION), whose negative coefficient indicates that reading time decreases if the word is fixated near the beginning or the end – thus reflecting the IOVP effect. Furthermore, we observe a small effect for LAUNCHDISTANCE. A smaller value of launch distance reflects a longer launch distance from the left. This is associated with longer reading times, as reflected in the negative coefficient, thus following expectations. It has been claimed that readers speed up while they move through a sentence (Ferreira and Henderson, 1993). Our data support this finding: we obtain a small negative coefficient for the position of the word within the sentence (SENTENCEPOSITION), which means later words are read faster.

For residual forward transitional probability (FORWARDTRANSITIONALPROBABILITY), we observed a negative coefficient. This is a bit harder to interpret, due to the fact that it doesn't relate to the transitional probabilities directly, but just to the part of the transitional probability which cannot be explained by unigram frequencies. When this transitional probability that goes beyond simple frequency is high, reading times are shortened, as reflected by the negative coefficient. This facilitation predicted by forward transitional probabilities is in line with McDonald and Shillcock's (2003b) results. However, McDonald and Shillcock (2003b) also find a negative coefficient for backward transitional probability, while in our data residual backward transitional probability (BACKWARDTRANSITIONALPROBABILITY) shows a small positive coefficient, which means that words which have a higher backward transitional probability (which can not be explained by unigram frequency or forward transitional probability) are predicted to lead to slightly longer reading times.

Predictor	Coefficient	Significance
(INTERCEPT)	241.83	***
WORDLENGTH	8.22	***
WORDFREQUENCY	-13.00	***
PREVIOUSWORDFREQUENCY	-6.24	***
PREVIOUSWORDFIXATED	-35.54	***
LANDINGPOSITION	-18.20	***
LAUNCHDISTANCE	-0.70	***
SENTENCEPOSITION	-0.24	***
FORWARDTRANSITIONALPROBABILITY	-1.97	***
BACKWARDTRANSITIONALPROBABILITY	1.03	***
INTEGRATIONCOST	-1.58	***
WORDLENGTH:WORDFREQUENCY	-2.93	***
WORDLENGTH:LANDINGPOSITION	-18.64	***

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 5.1: First pass durations for all words in the Dundee Corpus: coefficients and their significance levels for a model that includes all predictors as main effects and all binary interactions, minimised using the AIC.

While the coefficients for the non-syntactic predictors have plausible interpretations that are consistent with the previous literature on reading, the result for the integration cost predictor (INTEGRATIONCOST) is disappointing: we obtained a significant negative coefficient, which means that higher integration cost leads to shorter reading time, contrary to the prediction of DLT.

The same significant predictors were obtained when we ran mixed effects models for first fixation duration and in total reading times (we omit the tables here), with one exception: for first fixations, there was no effect of word length and no effect of integration cost.

One potential explanation for the lack of an effect of integration cost may be the fact that (following Gibson), we assumed identity as our integration cost function, i.e., I(n) = n. It is possible that there is a logarithmic relationship between integration cost and reading time (e.g., similar to that between frequency and reading time). We tested this by re-running the analysis reported in Table 5.1 with the integration cost function I(n) = log(n+1). However, again a significant negative coefficient for INTEGRATIONCOST

	(Intr)	Len	Frq	Frq-P	Fix-P	Dist	LPos	SPos	FTP	BTP	IC	L:Frq
Len	0.003											
Frq	0.005	0.453										
Frq-P	-0.018	-0.088	0.047									
Fix-P	-0.031	-0.086	0.018	0.423								
LPos	-0.020	0.126	-0.006	-0.062	-0.283							
Dist	0.010	0.069	0.033	-0.129	-0.405	0.198						
SPos	-0.001	-0.005	0.010	0.005	0.036	-0.036	-0.012					
FTP	0.004	0.235	0.157	-0.003	0.021	-0.026	0.006	-0.014				
BTP	0.003	0.138	0.118	-0.016	0.008	0.013	-0.013	-0.010	0.011			
IC	-0.002	-0.027	0.190	-0.009	-0.002	-0.007	0.003	-0.031	0.036	0.117		
Len:Frq	0.016	0.594	0.020	-0.073	-0.023	-0.017	-0.003	0.008	0.243	0.128	-0.057	
Len:Lpos	0.004	-0.436	0.085	0.003	-0.128	0.116	0.131	-0.011	-0.003	0.001	-0.020	-0.235

Table 5.2: Table of correlations between fixed effects (this is different from correlations between explanatory variables, which are reported in Table 5.7) for first pass durations for all words in the Dundee Corpus.

was obtained (though model fit improved slightly).

The model also contains two interactions: between word length and word frequency, and between word length and quadratic relative word landing position. The negative coefficient for the word length – frequency interaction means that words that are both long and frequent have slightly faster reading time, and correspondingly short infrequent words would have longer reading times than predicted by just word length and frequency alone. The negative coefficient for the word length – landing position interaction means that the IOVP effect is more extreme for long words: when a word is long and is fixated at the very beginning or end of the word, reading times are predicted to be shorter.

When we fitted mixed models for first fixation times and total times, we again found the same pattern of results as for first pass time, with the exception that the INTEGRATIONCOST effect was not significant in first fixations.

5.3.3 Discussion

In this experiment, we fitted mixed effects models on the reading times for all words in the Dundee Corpus, and found that integration cost is a significant negative predictor of reading time, i.e., that higher integration cost values correspond to shorter reading times, contrary to the prediction of DLT. This result can be explained by the fact that

DLT only provides a partial definition of syntactic processing complexity: integration costs are only assigned to nouns and verbs. All other words have an integration cost of zero, while there are very few nouns or verbs with an integration cost of zero (only non-head nouns in compounds).

We therefore further investigated the relationship between reading time and integration cost. We re-ran the mixed effects model in Table 5.1 on all words in the corpus and included integration cost as a factor, i.e., as a discrete predictor. When the DLT predictions are entered into the regression as categorical values, separate coefficients are estimated for each integration cost value.

These separate coefficients allow us to assess the influence of words with an integration cost of zero: the negative overall coefficient for integration cost as a continuous variable may be due to the fact that words with integration cost 0 are problematic, because not all of them may be covered by the theory. Therefore it is interesting to see whether there is an overall positive trend for words that are assigned an integration cost. Figure 5.1 plots integration cost values against their model coefficients and shows a general trend of higher integration cost values corresponding to larger coefficients (i.e., increased reading times), as predicted by DLT. The figure also shows that the coefficients for integration cost values one to nine are negative, i.e., the reading times for words with these integration cost values is shorter than the reading time for words with zero integration cost (which the model takes as the base value and assigns a coefficient of zero). This finding indicates that words with integration cost 0 can still generate difficulty, but that this difficulty is not captured by DLT, which only makes predictions for nouns and verbs. This result also means that the current coverage of DLT is clearly not sufficient for naturally occurring text. Most words in the corpus have integration cost values between zero and nine. In fact, the largest influence on the regression coefficient comes from words with integration cost 0 (approx. 125,000 fixated words) and integration cost 1 (approx. 84,000 fixated words). This explains why we found an overall negative coefficient of integration cost in Table 5.1 (where INTEGRATIONCOST was entered as a continuous predictor), even though higher integration cost values generally correspond to higher reading times in Figure 5.1.

As Figure 5.1 shows, the coefficient estimate for words with zero integration cost is higher than those of words with slightly higher integration cost. Since DLT traditionally only makes predictions for verbs and nouns, it would be interesting to find out at what other word types a similar cost might be incurred. To test whether some types of words take longer to read than others after factoring out low level effects, we computed

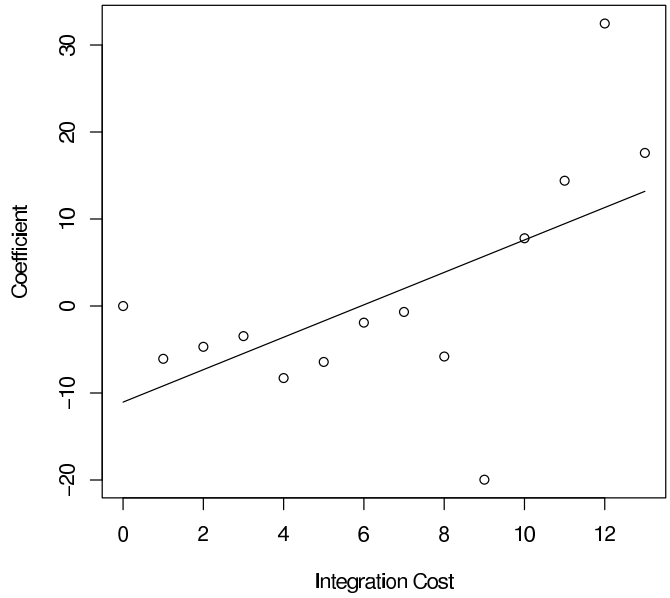


Figure 5.1: Coefficients for the factor integration cost in a mixed effects model on the words in the Dundee Corpus.

residual reading times on the Dundee Corpus by building a mixed effects model that contains all the non-syntactic predictors, and subtracted the reading times predicted by this model for the observed reading times. We analysed these data by partitioning them according to the words' parts of speech (POS). We found that adjectives, prepositions, sentence adjectives, and expletives have mean residual reading times larger than zero, which means they are read more slowly than would be expected according to word length, frequency, and the other non-syntactic predictors. The data therefore suggests that it could be interesting to extend DLT in a way that makes it possible to also assign an integration cost to those word categories.

5.4 Experiment 2: Integration Cost for Verbs and Nouns

In Experiment 1, a negative coefficient for integration cost was obtained when fitting a mixed effects model to predict reading times for all words in the Dundee Corpus. We concluded that this finding is due to the fact that DLT does not make integration cost predictions for words other than verbs and nouns. In the present experiment, we will explore this link further by providing a detailed analysis of integration costs for nouns and verbs.

5.4.1 Method

Data, statistical analysis, and implementation used were the same as in Experiment 1.

5.4.2 Results

Again, we will focus on results for first pass durations.

Nouns We first fitted a mixed effects model for the first pass durations for all the nouns in the Dundee Corpus (49,761 data points for the early measures, 57,569 data points for total durations) that included all predictors as main effects and all binary interactions, as well as intercepts and slopes under SUBJECT, minimised using the AIC. Integration cost was not a significant, positive predictor of reading time in this model.

When the data set was restricted further, viz., to nouns with non-zero integration cost (45,038 and 51,613 data points respectively), a significant, positive coefficient for integration cost was obtained. Furthermore, we found that model fit improves slightly when using the logarithmic integration cost function $I(n) = \log(n + 1)$ compared to when using a linear one. We further investigated why the effect of integration cost on nouns was only present if nouns with zero integration cost were excluded. This is particularly puzzling as it is rare that nouns receive an integration cost of zero; there is only way for this to happen in the corpus: the first word of noun-noun compounds and pronouns. We re-ran the model in Table 5.3, but included pronouns (an additional 4,840 data points for the early measures, 6,108 data points for total durations), despite their integration cost of zero. Again, a significant, positive coefficient of integration cost was obtained. First parts of compounds were relatively frequent in the Dundee corpus: there were 7,121 data points for total durations and 6,118 data points for the early measures; a large proportion of these cases consisted of proper names (such people's names or titles). We believe that these first parts of compound nouns must be responsible for the wrong integration cost estimations.

The coefficients of the model including nouns with integration cost greater than zero and pronouns are listed in Table 5.3. The significant positive coefficient for integration cost in this model means that nouns with higher integration cost take longer to read. As there seemed to be some collinearity between integration costs and frequency for nouns, we residualized integration cost. This did not change either the size or significance of the effect. The reported model also excluded outliers by automatically

Predictor	Coefficient	Significance
(INTERCEPT)	263.19	***
WORDLENGTH	10.78	***
WORDFREQUENCY	-16.64	***
PREVIOUSWORDFREQUENCY	-8.38	***
PREVIOUSWORDFIXATED	-47.25	***
LAUNCHDISTANCE	-0.35	*
LANDINGPOSITION	-27.57	***
SENTENCEPOSITION	-0.17	***
FORWARDTRANSITIONALPROBABILITY	-1.97	***
BACKWARDTRANSITIONALPROBABILITY	3.26	***
log(INTEGRATIONCOST)	7.12	***
WORDLENGTH:WORDFREQUENCY	-4.90	***
WORDLENGTH:LANDINGPOSITION	-15.43	***

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 5.3: First pass durations for nouns (with non-zero integration cost), and personal pronouns in the Dundee Corpus: coefficients and their significance levels for a model that includes all predictors as main effects and binary interaction, minimised using AIC.

excluding all data points with high leverage from the model.

We fitted mixed models for first fixation durations and total times, and found the same set of significant predictors, with the following exceptions: for first fixations, there was no significant effect of WORDLENGTH, and the effect of INTEGRATIONCOST was small, and there were no significant interactions. However, we did find a significant positive effect for integration cost in the total times analysis.

Verbs Just as for nouns, we fitted a mixed effects model for the first pass durations for all the verbs in the Dundee Corpus (the model again included all main effects and all binary interactions). No significant, positive coefficient for integration cost was obtained in this model. We re-ran the model with verbs that exhibit a non-zero integration cost, and with a logarithmic instead of a linear integration cost function. Again, integration cost was not a significant, positive predictor of reading time.

We then fitted a model that included the part of speech of the verb as a predictor. The rationale behind this is that verb reading time may differ by part of speech, e.g., inflected verbs are read more slowly than infinitives. This model only included verbs

with non-zero integration costs and used a logarithmic integration cost function. We found that this time, integration cost was a significant, positive predictor of reading time (though the size of the coefficient was smaller than for nouns).

In order to further investigate the integration cost effect that we found for verbs, we computed residual reading times for this data set. On the residuals, we then fitted a model that includes a predictor that indicates the part of speech of the dependent that is integrated at a given verb (or sequence of parts of speech if multiple dependents are integrated). The coefficients in this model indicate which dependents lead to higher or lower integration costs, see Table 5.4. We observe that, as predicted by DLT, the integration of nouns (parts of speech NN, NNP, NNS) or adverbs (part of speech RB) leads to longer reading times, unless there is also an auxiliary (AUX) that occurs before the verb. The auxiliary thus seems to facilitate integration of nouns at the verb.

5.4.3 Discussion

In Experiment 1, we saw that DLT integration cost does not constitute a broad-coverage theory of syntactic complexity, in the sense that integration cost failed to emerge as a significant, positive predictor of reading time on the whole of the Dundee Corpus. We hypothesised that this is due to the fact that DLT only makes partial integration cost predictions, viz., for nouns and verbs only. In the present experiment, we investigated this further by analysing the performance of DLT on verbs and nouns in more detail.

We showed that integration cost is a significant, positive predictor of reading time on nouns with a non-zero integration cost, and thus supports the hypotheses in DLT. However, this result reflects only effects on a small amount of the data: In its standard form (Gibson, 2000), DLT does not make very interesting predictions for nouns. Most nouns have an integration cost of 1, because a discourse referent is built. The only cases in which nouns can receive an integration cost greater than 1 are in constructions such as *request for permission*, where *permission* is analysed as the head of the NP, genitive constructions like *the Nation's criminals*, and copula constructions. In the latter, nouns are considered to be the head of the phrase and integrate the verb *be*. This means that the integration cost for the noun depends on the number of discourse referents intervening between the noun and *be*.

We also investigated the two cases in which DLT assigns an integration cost of zero to nouns. The first case is pronouns, which DLT assumes to constitute old discourse referents, not incurring a cost. We extended our model by including pronouns (as

Dependents	Coefficient	Significance	N
PRP-AUX-NN	-81.45	**	15
PRP-AUX	-76.24	**	13
NNP-AUX-AUX	-62.41	**	21
RP	-62.34	*	12
NNP-AUX	-59.52	*	17
PRP-MD	-56.44	*	17
NNS-AUX-AUX	-35.65	*	57
NNS-MD-AUX	-30.75	**	110
PRP-AUX-PRP-AUX	-29.72	***	184
NN-MD-AUX	-25.35	**	153
PRP-AUX	-22.64	***	700
PRP-AUX-RB	-21.75	*	133
AUXG	-20.26	*	121
NNP-AUX	-19.05	**	301
TO-PRP	-16.97	***	723
NNP	12.01	**	1372
NN-RB	22.26	*	127
AUX-NNP	66.11	*	15
VBP	67.69	*	10
RB	75.88	**	15
NN-NNS	76.43	***	25
PRP-MD-PRP-MD-JJ	105.4	*	65

Table 5.4: First pass durations for verbs (with non-zero integration cost) in the Dundee Corpus: coefficients for the verbal dependents and their significance levels for a model fitted on residual reading times. Abbreviations in the table refer to part of speech tags used by the Penn Treebank annotation: AUX: auxiliary, PRP: personal pronoun, NN: singular or mass noun, NNP: proper noun, singular, RP: particle, MD: modal, NNS: plural noun, RB: adverb, AUXG: auxiliary present participle, TO: preposition *to*, JJ: adjective, VBP: non-third person singular present verb.

the only nouns with zero integration cost), and still found that integration cost was a significant, positive predictor, which provides evidence for the DLT assumption that pronouns carry zero integration cost. The second case of zero integration cost is noun-noun compounds, for which DLT assumes that the first noun incurs no integration cost. However, when we fitted a model on all nouns (including the ones with zero integration cost), we failed to obtain a significant coefficient for integration cost. This indicates that the DLT assumption of cost-freeness for the first noun of a noun-noun compounds is incorrect. Rather, we have to assume that a discourse referent is already being established when the first noun in the compound is encountered, i.e., this noun should incur a non-zero cost.

At this point, it becomes important which version of DLT is used to compute integration cost values. In contrast to the Gibson (2000) version used in this thesis, the Gibson (1998) version of DLT assigns higher integration costs to nouns that occur after their head noun. In order to test how crucial this assumption is, we implemented the 1998 version and conducted the same experiments as with the 2000 version, but this failed to yield an improved fit on our data set.

For either version, we observed that DLT only makes a restricted range of predictions for nouns: with few exceptions, all head nouns are assigned an integration cost of 1. Arguably, this limits the power of the theory in explaining reading time data for noun phrases in a corpus, which are often complex. This problem could be addressed by extending DLT along the lines suggested by Warren and Gibson (2002). They provided evidence that processing complexity at the verb varies with the referential properties of the NP to be integrated, as expressed by the NP's position on the Givenness Hierarchy (Gundel et al., 1993). Warren and Gibson (2002) find that complexity increases from pronouns to names to definite NPs to indefinite NPs and therefore suggest that a continuous integration cost metric needs to be developed that takes the givenness status of the integrated NP into account. This would result in a wider range of integration cost values for the nouns in the Dundee Corpus, potentially making it possible to explain more variance in the reading time record.

In addition to looking at nouns, we also investigated the relationship between reading times and integration cost for verbs and were able to show that integration cost is a significant positive predictor of verb reading times. This result was only obtained for a model that includes the parts of speech of the verbs as an additional predictor. This indicates that integration cost can predict processing difficulty for verbs, but that this effect is variable across parts of speech.

As verb integration cost is at the heart of DLT (which predicts only limited variation in noun integration cost, see above), we investigated this result further. We fitted a model on the residual reading times that included the parts of speech of the dependents to be integrated at the verb as a predictor. This analysis revealed the following pattern (see Table 5.4): positive coefficients were obtained for the integration of nominal dependents (indicating that this integration leads to increase reading time), while negative coefficients were obtained for the integration of auxiliaries (which means that this integration decreases reading time). This result has an interesting implication for DLT. On the one hand it confirms the DLT assumption that an integration cost is incurred at the verb when nominal dependents are integrated. On the other hand, it shows that this does not extend to cases where an auxiliary precedes the main verb. A possible explanation is that the relevant integration cost is not incurred at the main verb, but at the auxiliary itself, which integrates nominal dependents and thus incurs a non-zero integration cost (DLT assumes that auxiliaries are cost-free). When the auxiliary is then integrated with the main verb, it facilitates integration (hence the negative coefficient), as the main work of the integration of the nominal dependents has already happened at the auxiliary. Note that this assumption is compatible with syntactic theories such as Head-driven Phrase Structure Grammar (Pollard and Sag, 1994), which assume that auxiliaries inherit the subcategorization frame of the main verb, and that dependents are unified (integrated) into the subcategorization frame at the auxiliary. In this context, it is interesting to note that Warren and Gibson (2002) found a reading time effect for auxiliaries. Auxiliaries following definite NPs were read more slowly than auxiliaries following pronouns. This result in consistent with our findings in the Dundee Corpus, i.e., that auxiliaries, and not just main verbs, show integration cost effects. However, Warren and Gibson (2002) interpret their finding as a spillover effect. Clearly, more experimental work would be needed to test the effect of auxiliaries on reading times on the main verb, and integration effects on auxiliaries.

5.5 Experiment 3: Surprisal

Experiments 1 and 2 indicate that there is evidence that DLT integration cost is a predictor of reading time in the Dundee Corpus. However, DLT cannot be regarded as a broad coverage model, as we found integration cost effects only when the our model was limited to verbs and certain nouns. The present experiment has the aim of evaluating Surprisal as an alternative model of syntactic processing complexity. Unlike

DLT, Surprisal is designed to make predictions for all words in a corpus, on the basis of a probabilistic grammar. We will test two versions of Surprisal (lexical and structural), and compare them against non-syntactic probabilistic predictors of reading time (forward and backward transitional probability).

5.5.1 Method

Data and statistical analysis were the same as in Experiments 1 and 2. For calculating the Surprisal values for the words in our corpus, we parsed the Dundee Corpus with an incremental parser which returns a prefix probability for each word in the corpus, i.e., the probability in Equation (2.5) from the Section 2.2.4, here repeated as equation (5.1) for convenience:

$$P(w_1 \cdots w_k) = \sum_T P(T, w_1 \cdots w_k) \tag{5.1}$$

$$S_{k+1} = -\log \frac{P(w_1 \cdots w_{k+1})}{P(w_1 \cdots w_k)} = \log \sum_T P(T, w_1 \cdots w_k) - \log \sum_T P(T, w_1 \cdots w_{k+1}) \tag{5.2}$$

We can then use Equation (2.6), here repeated as Equation (5.2) to obtain the Surprisal value for a word w_{k+1}: we subtract the logarithmic prefix probability for w_{k+1} from the logarithmic prefix probability for w_k. The parser used was Roark's (2001a) incremental top-down parser. This is a probabilistic parser trained on Sections 2 – 21 of the Penn Treebank (Marcus et al., 1993), a corpus of English text from the Wall Street Journal which has been manually annotated with phrase structure trees. The parser achieves a broad coverage of English text and has a precision and recall of 85.7% for labelled brackets (Roark, 2001a). As the Dundee Corpus also consists of newspaper text, we expect a similar performance on the Dundee Corpus.

Again, there was a mismatch between tokenization of the parser and the Dundee Corpus. Just as for integration cost, we decided to combine Surprisal predictions by addition. Surprisal captures the amount of probability mass of analyses that are not compatible with the current input given the prefix. Two words which are one token in the Dundee corpus (like *we'll*) carry the same information as two separate adjacent tokens (*we* and *'ll*, and thus rule out the same structures, such that the Surprisal of *we'll* is exactly the same as the Surprisal of *we* plus the Surprisal of *'ll* (see Equation (5.3)).

$$S_{k+1} + S_{k+2} \quad = \quad -\log P(w_{k+1}|w_1 \cdots w_k) + -\log P(w_{k+2}|w_1 \cdots w_{k+1}) \tag{5.3}$$

$$
\begin{aligned}
&= -\frac{\log P(w_1 \cdots w_{k+1})}{P(w_1 \cdots w_k)} - \frac{\log P(w_1 \cdots w_{k+2})}{P(w_1 \cdots w_{k+1})} \\
&= -\log P(w_1 \cdots w_{k+1}) + \log P(w_1 \cdots w_k) - \\
&\quad \log P(w_1 \cdots w_{k+2}) + \log P(w_1 \cdots w_{k+1}) \\
&= \log P(w_1 \cdots w_k) - \log P(w_1 \cdots w_{k+2}) \\
&= -\log \frac{P(w_1 \cdots w_{k+2})}{P(w_1 \cdots w_k)} \\
&= -\log P(w_{k+1}, w_{k+2} | w_1 \cdots w_k) \\
&= S_{k+1,k+2}
\end{aligned}
$$

Surprisal was estimated in two different ways. The first version was fully lexicalized, i.e., it takes into account the exact words of a string when calculating structural and lexical probabilities. This lexicalized version was obtained using the default configuration of the Roark parser. The second version was not lexicalized, i.e., it only used the structural probabilities. This structural model (also described in (Roark et al., 2009)) does not take into account word frequency or the probability of a word being assigned a specific POS tag (i.e., there are no lexical rules of type $V \rightarrow$ *wrote*). This structural version of Surprisal helps us to factor out frequency effects.

5.5.2 Results

Table 5.5 shows the coefficients and significance levels obtained when running a mixed effects model on first pass durations in the Dundee Corpus. As in Experiment 1, this model was computed over all words in the corpus, and included all non-linguistic predictors as well as lexical Surprisal (LEXICALSURPRISAL), structural Surprisal (STRUCTURALSURPRISAL), and forward and backward transitional probability.

Table 5.5 shows that structural Surprisal is a significant, positive predictors of reading time (high Surprisal leads to longer reading time). The coefficient for STRUCTURALSURPRISAL is small, but this has to be interpreted in the context of the range of this predictor: the values for structural Surprisal range from 0 to 16, with a mean Surprisal of 2.4.

Residualized lexical Surprisal (LEXICALSURPRISAL) has a negative coefficient in Table 5.5, which means that the proportion of lexical Surprisal which is not captured in either unigram frequency of a word, transitional probability or structural Surprisal made incorrect predictions of reading times: larger residual lexical Surprisal is equated to faster reading by the regression model. The same effect was also found in a model without structural surprisal, where lexical surprisal was only residualized with respect

to unigram frequencies and forward transitional probabilities. Residualized forward transitional probability (i.e. the part of forward transitional probability that cannot be explained by simple word frequencies) was a significant negative predictor of reading time (higher probability means lower reading time), thus satisfying expectations. As detailed in Section 5.2, forward transitional probability can be regarded as a simple form of Surprisal that only takes into account the immediate context (the preceding word). Residual backward transitional probability has a positive coefficient.

We fitted mixed effects models for total times, which also showed a positive effect of structural Surprisal and a negative effect of residualized lexical Surprisal. Structural Surprisal has a larger coefficient in the total time model.

In the first fixation model, the interaction between word length and frequency did not come out as a significant predictor of reading times. Furthermore, WORDLENGTH, BACKWARDTRANSITIONALPROBABILITY and STRUCTURALSURPRISAL were not significant predictors for first fixation times. On the other hand, FORWARDTRANSITIONALPROBABILITY was attributed a larger negative coefficient in the first fixation model.

The same models were also fitted with random slopes (including one each for lexical and structural Surprisal) under subject. Results were, as in Experiment 1, very similar, with significance values generally being a bit lower for all predictors, and similar high correlations. Section 5.6 reports models including random slopes under subject for all three reading measures, where the correlation problem is solved.

5.5.3 Discussion

This experiment showed that Surprisal can function as a broad-coverage model of syntactic processing complexity: we found that structural Surprisal was a significant, positive predictor of reading time on arbitrary words in the Dundee Corpus. This sets Surprisal apart from integration cost, which does not make predictions for all words in the corpus, and for which we only obtained significant effects on verbs and nouns.

We failed to find a corresponding effect for lexical Surprisal. This indicates that forward transitional probability and structural Surprisal taken together are better predictors of reading times in the Dundee Corpus than lexical Surprisal, which combines these two components. Forward transitional probability can be regarded as a simple approximation of Surprisal (see Section 5.2), and our results indicate that this approximation is sufficient, at least when it comes to predicting the reading times in the corpus.

Predictor	Coefficient	Significance
(INTERCEPT)	241.96	***
WORDLENGTH	8.40	***
WORDFREQUENCY	-12.49	***
PREVIOUSWORDFIXATED	-35.55	***
FREQOFPREV	-6.12	***
LANDINGPOSITION	-18.22	**
LAUNCHDISTANCE	-0.70	***
SENTENCEPOSITION	-0.24	***
FORWARDTRANSITIONALPROBABILITY	-1.73	***
BACKWARDTRANSITIONALPROBABILITY	1.00	***
STRUCTURALSURPRISAL	0.49	***
LEXICALSURPRISAL	-0.63	***
WORDLENGTH:WORDFREQUENCY	-2.87	***
WORDLENGTH:LANDINGPOSITION	-18.78	***

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 5.5: First pass durations for all words in the Dundee Corpus: coefficients and their significance levels for a model that includes all predictors as main effects, and all binary interaction, minimised using the AIC.

Structural Surprisal, on the other hand, takes structural probabilities into account, but disregards lexical probabilities, and therefore is a significant predictor of reading time, even if forward transitional probability is also entered into the model. We conclude that structural Surprisal is able to explain a component in the reading time data that neither transitional probabilities, nor any of the other non-syntactic predictors can explain. This is evidence for Hale's (2001) and Levy's (2008) hypothesis that the incremental disconfirmation of syntactic hypotheses by the parser can explain processing complexity.

Our Surprisal results are corroborated by a number of later pieces of work. Ferrara Boston et al. (2008) found that structural Surprisal is a significant predictor of reading times on the Potsdam Sentence Corpus. The Potsdam Sentence Corpus differs in a number of ways from the Dundee corpus: it uses a different language (German) and it consists of unconnected sentences, which were manually constructed for experimental purposes, rather than taken from naturally occurring text. Also, it

is smaller in terms of items (144 sentences), but larger in terms of participants (272 participants) than the Dundee corpus. It is therefore encouraging that our results are consistent with Ferrara Boston et al.'s (2008), in spite of these corpus differences. Ferrara Boston et al. (2008) did not test lexical Surprisal or integration cost on their data set, but they compared two versions of structural Surprisal, estimated either using a context-free grammar (i.e., in the same way as in the present study), or using a dependency grammar. In both cases, the Surprisal estimates were a significant predictor of reading times. Furthermore, Roark et al. (2009) also found that structural Surprisal calculated by the same parser as used in this thesis correctly predicts reading times on the Bachrach corpus. Additionally, Frank (2009) found a significant positive effect of structural Surprisal on the Dundee corpus, both using the Roark parser and using an SRN. Both of Frank's models were trained on POS-tag sequences.

5.6 Experiment 4: A Comparative Model of DLT and Surprisal

5.6.1 Method

To give a comparative overview of the syntactic predictors discussed in this chapter and to address issues of collinearity in models including random slopes, we fitted a baseline mixed effects model that includes only the predictors which are not of interest, their interactions and random slopes under subject, and then fitted separate models for the three predictors we are interested in, integration cost, lexical and structural Surprisal, on the residuals of the baseline model. The advantage of this method is that all predictors are fitted on the exact same data, and that there are no possible effects of collinearity on the effects we're interested in. Furthermore, we do not need to separately residualize syntactic predictors against other correlated predictors.

5.6.2 Results

The results for the baseline model as well as the models on its residuals are given in Table 5.6. We will start by discussing the columns for first pass times, which showed that integration cost, structural Surprisal and lexical Surprisal are all significant predictors of reading time. However, the coefficient of INTEGRATIONCOST was negative, confirming that integration cost is not a broad-coverage predictor of reading time (as

	First Fix		First Pass		Total Time	
Predictor	Coef	Sig	Coef	Sig	Coef	Sig
(INTERCEPT)	205.50	***	241.18	***	254.07	***
WORDLENGTH	0.71	*	8.11		7.36	***
WORDFREQUENCY	-6.33	***	-12.34	***	-15.80	***
PREVIOUSWORDFREQUENCY	-3.11		-6.19	*	-6.35	***
PREVIOUSWORDFIXATED	-10.95	***	-33.66	*	-35.60	***
LAUNCHDISTANCE	-1.63	***	-0.75		-0.86	
LANDINGPOSITION	8.31	***	-18.00		-21.39	***
SENTENCEPOSITION	-0.05	**	-0.24	***	-0.28	***
FORWARDTRANSITIONALPROB	-1.59	***	-1.97		-2.77	***
BACKWARDTRANSITIONALPROB	0.71	*	1.18		1.36	**
WORDLENGTH:WORDFREQUENCY	-1.15	***	-3.06	***	-4.15	***
WORDLENGTH:LANDINGPOSITION	rem	–	-19.21	***	-18.59	***
INTEGRATIONCOST	-0.18		-1.72	***	-2.82	***
LEXICALSURPRISAL	-0.04		-0.15	*	-0.16	
STRUCTURALSURPRISAL	0.11		0.56	**	1.21	***

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 5.6: Coefficients and significance levels for models of first fixation times, first pass durations, and total reading times for all words in the Dundee Corpus. The models include all predictors that are not of primary interest, interactions between them, and their slopes under subject. Predictors of interest and their random slopes under subject were run in separate models on the residuals of the basic model. Predictors marked "rem" were removed from the regression as they did not significantly reduce the AIC.

shown in Experiment 1). Furthermore, LEXICALSURPRISAL has a small negative coefficient, meaning that words with higher lexical Surprisal show longer reading times, which is not what is expected according to Surprisal theory (but in line with the finding of Experiment 3). Note however, that the negative effect, after outlier removal, was not significant for total reading times. Finally, STRUCTURALSURPRISAL is confirmed as a significant positive predictor of first pass reading times.

Turning to the results for first fixation times (see Table 5.6), we again found a significant negative effect of forward transitional probability, and a small positive effect of backward transitional probability. The interaction between word length and landing

	INTEGRATION COST	WORD FREQ	LEXICAL SURPRIS	STRUCT. SURPRIS	FORWTRANS PROB	
WORDFREQUENCY	-0.25					
LEXICALSURPRIS	0.17	-0.57				
STRUCT.SURPRIS	-0.07	0.04	0.36			
FORWTRANSPROB	-0.20	0.62	-0.66	-0.10		
BACKTRANSPROB	-0.26	0.62	-0.53	0.04	0.68	

Table 5.7: Correlation coefficients (Pearson's r) between the predictors, for fixated words ($N = 237,163$).

position was removed from the model, as it did not improve model fit, and word length turned out to be a much smaller, just marginally significant predictor. None of the syntactic predictors significantly improved model fit.

The results for total reading times (see also Table 5.6) largely replicated the results for first pass times; again integration cost, and structural Surprisal were significant predictors. However, the effect of lexical Surprisal (which had wrong polarity anyway) failed to reach significance. The coefficient for integration cost was negative, also replicating the findings for first pass times. Note that LAUNCHDISTANCE and LANDINGPOSITION only reflect data from the first fixation on the word. We tried to remove it from the model for this reason, but model fit got significantly worse as a result, so we decided to leave it in.

5.6.3 Discussion

Results for regression on residuals for the sentence processing measures shows that the effects found are very stable also under this conservative way of measuring. Estimating each of the syntactic predictors on the residuals of the same baseline model also brings up the question of how similar the predictors integration cost, structural Surprisal and lexical Surprisal are to each other. Do they capture overlapping parts of the variance in the data or not? Would one predictor explain away the others? To address this issue, we computed correlations between integration cost and the different incarnations of Surprisal (lexical and structural Surprisal, forward and backward transitional probabilities), and word frequency. The result is given in Table 5.7; all correlations are statistically significant except for the pair WORDFREQUENCY–STRUCTURALSURPRISAL (even small correlations are significant due to the large number of observations).

As expected, forward and backward transitional probability are highly correlated. Furthermore, the lexicalized measures (lexical Surprisal and transitional probabilities) are highly correlated with word frequency. The high correlation between lexical Surprisal and forward transitional probability confirms the intuition that these two measures are in fact both incarnations of Surprisal, but of a different level of granularity. On the other hand, structural Surprisal is not significantly correlated with the other measures, including word frequency (though there is a weak correlation with lexical Surprisal). This confirms that structural Surprisal really captures structural probability effects, without taking lexical probabilities into account. Crucially, Table 5.7 also shows that integration cost is orthogonal to Surprisal and the other frequency-based predictors: there is no strong correlation between INTEGRATIONCOST and any of the other predictors. This finding holds even if we compute correlations only for the verbs in the Dundee Corpus (not shown in the table): the correlation between integration cost and structural Surprisal is approximately 0.05 for verbs, while the correlation between integration cost and lexical Surprisal is approximately 0.01 for verbs. This confirms that integration cost and Surprisal are orthogonal: if there was a relationship between them, it should manifest itself on verbs, as verbs are the words with the largest variation in integration cost (compared to nouns, which mostly have an integration cost of one, and the other words, which have an integration cost of zero; see also Section 5.4.3).

The lack of correlation taken together with the fact that significant effects are found both for Surprisal and DLT integration cost is supporting evidence for our hypothesis that both DLT and Surprisal capture relevant aspects of processing difficulty, but that these aspects are complementary. This result suggests that a complete theory of sentence processing complexity needs to include two mechanisms: a backward-looking one as proposed by DLT, and a forward-looking one as proposed by Surprisal. When a new word is processed it incurs two types of processing cost: the cost of integrating previous material with the new word, and the cost of discarding alternative syntactic predictions that are not compatible with the new word. The first type of cost corresponds to locality effects that have been observed extensively in the literature (see Gibson, 1998). The second type of cost corresponds to anti-locality effects which have been reported recently (Konieczny, 2000; Vasishth and Lewis, 2006). In order to capture both types of cost (and yield broad-coverage results on an eye-tracking corpus), it is necessary to develop a unified model that encompasses both the prediction of upcoming material and the subsequent verification and integration processes, as the one described in the later chapters of this thesis.

5.7 General Discussion

An important point to consider in this evaluation is the fact that the predictions of both DLT and Surprisal depend on the grammar formalism that they are operating on. In DLT, syntactic structures (head–dependent relations) determine the amount of integration cost that is incurred by a given sequence of words. While there are many clear cases of what constitutes the head, the dependent and the relation between them can be subject to debate in the linguistic literature. We assume here that the dependency structures output by MINIPAR forms the basis of the integration cost computations (see Section 5.3.1.3). MINIPAR uses one particular codification of dependency grammar (Sampson, 1995), and it is therefore conceivable that our results would change if we computed integration cost using a parser that makes a different set of representational assumptions.

It is important to note that Surprisal also requires representational assumptions. The definition of Surprisal in Equation (2.4) does not mention syntactic structures explicitly. However, in order to compute the conditional probability in this equation, prefix probabilities have to be obtained, which requires summing over all possible analyses of a string. The number and type of these analyses will differ between grammatical frameworks, which entails that representational assumptions do play a role for Surprisal. In this work, we only investigated the predictions of one type of syntactic representations, viz., those of Roark's (2001a) parser, which generates Penn Treebank-style structures. It is possible that other syntactic models will yield different Surprisal estimates and fit the reading time data more closely, or model different aspects of the data. (This has been investigated by Ferrara Boston et al. (2008), who compare dependency and phrase-structure versions of Surprisal.)

Apart from its theoretical contribution, this chapter also makes a methodological contribution to the field. To the best of our knowledge, the work described here was the first time that theories of sentence processing have been tested on broad-coverage data extracted from an eye-tracking corpus. Since the method of evaluation on eye-tracking corpora has subsequently been adopted by a number of people in the research community (Ferrara Boston et al., 2008; Frank, 2009; Wu et al., 2010). We believe that the corpus-based approach presented here constitutes an important new method for evaluating models of sentence processing. Such models are currently tested exclusively on data obtained for isolated, artificially constructed sentences in controlled lab experiments. The validity of the models can be enhanced considerably if we are able to show

that they scale up to model reading data from an eye-tracking corpus, which contains naturally occurring, contextualised sentences. Furthermore, the use of eye-tracking corpora has the advantage of convenience and flexibility: it makes it possible to study arbitrary syntactic constructions, provided that they occur sufficiently frequently in the corpus. There is no need to run a new experiment for every construction or every hypothesis to be investigated.

While the corpus-based approach has great potential, there are limitations as well. The fact that naturally occurring sentences are used means that it is much more difficult to control for confounding factors. We have attempted to include all potentially confounding factors as co-variates in mixed effects models, thus controlling for the influence of these factors. However, it is possible that there are some confounds that we have failed to identify, and therefore they could introduce artefacts in our models. In an experimental setting, the experimenter will often construct materials so that they are matched across conditions, i.e., the sentences only differ in the aspects that the experimenter wants to manipulate, an are identical in all other ways. This reduces the possibility that there are confounding factors that have not been taken into account. Another limitation of the corpus-based approach is data sparseness. No corpus can be so big that it contains all syntactic structures that an experimenter might want to get data on. For example, if we want to investigate prepositional phrase attachment, then there is a good chance that there are enough relevant sentences in the Dundee Corpus. If we want to investigate doubly nested relative clauses, on the other hand, then probably there are not enough tokens. This situation is even worse if we want to study structures that are ungrammatical or cause serious processing disruption. These probably do not occur in the corpus at all. To summarise, experimental data and corpus data have complementary strengths and weaknesses, and should be used in conjunction to maximise the evidence for or against a given theoretical position.

5.8 Conclusions

In this chapter, two theories of syntactic processing complexity were evaluated against reading time data extracted from a large eye-tracking corpus: Gibson's (1998; 2000) Dependency Locality Theory (DLT) and Hale's (2001) Surprisal. The goal of the study was to investigate whether the two theories provide accurate predictions for arbitrary words in naturalistic, contextualised text (as opposed to artificially constructed experimental materials, presented out of context and repeated many times).

We found that DLT's integration cost was not able to provide reading time predictions for the Dundee corpus as a whole. This was largely due to the fact that DLT only assigns integration cost values to verbs and nouns; this means that the majority of words in the corpus have an integration cost of zero. However, we were able to show that integration cost is a significant predictor of reading time if the verbs and nouns in the corpus are analysed separately. When we tested DLT predictions against the verbs in the Dundee corpus, we found evidence that the integration cost definition for auxiliaries needs to be revised: verbs that integrate an auxiliary and a nominal dependent exhibit a reduced difficulty compared to verbs that only integrate a nominal dependent. For nouns, we found that compounds need to be investigated further – our data suggests that integration cost might already occur at the first noun component in a compound, and not just at the head as current theories would assume.

In the second part of this chapter, we evaluated the predictions of Hale's (2001) Surprisal measure on the Dundee corpus. We computed Surprisal in two ways: lexical Surprisal was estimated using a probabilistic parser that utilises lexical (word-based) probabilities as well as structural (rule-based) probabilities. We found that only structural Surprisal was a significant positive predictor of reading times. This result shows that structural Surprisal is a good candidate for a broad-coverage model of syntactic processing complexity; it generates accurate numerical predictions for all types of words in the corpus, not just for nouns and verbs, as integration cost does.

Our findings regarding lexical Surprisal indicate that a fully lexicalized parsing model does not offer an advantage over a structural one. However, this does not mean that there is no role for lexical information in modelling reading times. The experimental literature offers broad evidence for the fact that sentence processing relies on lexical information, such as subcategorization frame frequencies (e.g., Garnsey et al. (1997); Trueswell et al. (1993)) and thematic role preferences (e.g., Garnsey et al. (1997); Pickering et al. (2000)). Recent probabilistic models of human sentence processing have attempted to integrate such information with the structural probabilities generated by a parser (Narayanan and Jurafsky, 2002; Padó, 2007). It seems likely that these models (which are effectively structural model augmented with a limited form of lexical information) would yield a more accurate account of reading times in the Dundee Corpus.

A central finding of the last experiment was the fact that Surprisal and integration cost are uncorrelated, both for arbitrary words in the corpus, and for verbs (for which DLT makes the bulk of its predictions). This result suggests that a complete theory

of sentence processing complexity needs to include two mechanisms: a backward-looking one as proposed by DLT, and a forward-looking one as proposed by Surprisal. The next chapter of this thesis proposes a way to combine these aspects, while also observing the psycholinguistically motivated mechanisms of incrementality and prediction in human sentence processing.

Chapter 6

A Theory of Explicit Predictions

This chapter proposes a new theory of human sentence processing which explicitly models prediction during language comprehension.

The first part of this chapter explains the theory's underlying assumptions, incrementality, full connectedness, prediction and verification. The theory is conceptualised as a ranked parallel processor and aims to make predictions that cover all types of words. The theory furthermore provides a natural way of accounting for both a forward and a backward looking process, which bear similarities to surprisal and integration cost. Section 6.2 outlines the linking theory, which derives processing difficulty predictions from the parsing process. Section 6.3 discusses which of the existing grammar formalisms can be used or adapted most easily to model the assumptions made by the theory, in order to be used as a basis for a parser which can automatically generate syntactic analyses that follow the outlined assumptions.

Parts of this chapter have been first presented at CUNY 2008, AMLaP 2008 and TAG+9 2008.

6.1 Fundamental Assumptions and Properties

The fundamental assumptions of the proposed theory are strictly incremental processing with full connectivity, prediction in combination with a verification mechanism, and ranked parallel processing.

6.1.1 Incrementality

There are different interpretations of what "incremental processing" on the syntax level means. The most general interpretation is that it involves left-to-right processing on a word by word basis. But then the question arises, how "complete" that processing has to be – are the syntactic relations between the all processed words fully specified even if final evidence for dependencies can be expected to only occur later on in the sentence? In the less strict interpretation of incremental processing, words can be partially connected and these partial structures stored on a stack until further evidence for how to connect them is encountered. The strongest form of incrementality, which we will refer to as *strict incrementality* or *full connectedness* entails that all words which have been preceived so far are connected under the same syntactic root node, which means that all relations have been specified completely (of course, competing analyses can exist in parallel).

In this section, we will review evidence for full connectedness. First, we summarise the discussion in the literature, which took place in the early 90's, about incremental interpretation and the relationship of syntax and semantics, also known as the *strict competence hypothesis*. We then review empirical results from psycholinguistic research about the degree of incrementality in human sentence processing.

6.1.1.1 Incremental Interpretation and Strict Competence

One of the most fundamental questions in the design of a theory of syntactic processing concerns the relationship between syntactic processing and semantic processing, because many psycholinguistic experiments observe only *incremental interpretation*, which means that the *semantics* of the partial sentence has been composed based on the words that have been perceived at a certain point. Claims about syntax are based on c-command relations, as in (Sturt and Lombardo, 2005), role assignment (Kamide et al., 2003), pre-head garden pathing in head final languages (Aoshima et al., 2004) or co-reference and binding (Aoshima et al., 2007; Yoshida et al., 2009), see below.

In the literature, there is an extensive discussion of whether it is necessary to assume syntactic connectedness in order to achieve incremental interpretation. Steedman (2000) argues for the *strict competence hypothesis* which essentially means that only syntactic constituents can receive an interpretation, on the grounds that it would be necessary to assume a more complex design of the human processing system if it had separate mechanisms for dealing with incomplete structures. Note however that the no-

tion of constituents is more general in Steedman's work than in most standard linguistic definitions. It refers to constituents licensed by Combinatory Categorial Grammar (CCG), where each syntactic constituent can be directly linked to a semantic interpretation.

On the other hand, Shieber and Johnson (1993) argue that even a bottom-up parser where syntactic structure has not yet been connected, can be sufficient for incremental interpretation. They showed this using Synchronous Tree Adjoining-Grammar (STAG) as a formalism, which constructs syntactic and semantic analyses simultaneously. However, in order to produce the necessary information to create the semantic relationships not yet expressed in the syntax, additional machinery is needed.

A simpler model requires the syntactic relationships to be established in order for the semantic ones to be made, hence tightly coupling the syntactic and semantic processing, as e.g. in CCG. In this thesis, we assume the simpler relationship, where all observed incremental interpretation is based on connected syntactic structures.

6.1.1.2 Experimental Evidence Supporting Incrementality with Full Connectedness

Recent 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. This is a fact that any cognitively motivated model of language understanding should capture. Full connectedness is a stronger claim. It means that all words must be connected into a syntactic structure at any point in the incremental processing of a sentence. Evidence for full connectedness comes from findings such as the one presented by Sturt and Lombardo (2005), see Example (1).

(1) a. The pilot embarrassed John and put himself in an awkward situation.
b. The pilot embarrassed Mary and put herself in an awkward situation.
c. The pilot embarrassed John and put him in an awkward situation.
d. The pilot embarrassed Mary and put her in an awkward situation.

The experimental items are constructed in order to test for a gender default mismatch effect in condition (1-b), where the pronoun herself refers back to *the pilot*. If people have connected all parts of the syntactic structure completely at this point, the c-command relation between the *pilot* and the pronoun should be established. In the experiment, the gender mismatch effect occurs directly when the reflexive pronoun

is encountered (and not just at the end of the sentence), suggesting that the syntactic c-command relation link must have been created at the point of processing *herself*. Conditions (1-c) and (1-d) were included to rule out a structurally-blind strategy for connecting the pronoun, in which the pronoun would be connected to the first noun in the sequence.

More evidence for connectedness comes from visual world studies like the one by Kamide et al. (2003). In their study, participants listened to sentences like the ones shown in Example (2) while looking at a visual scene that included four objects, three of which were in a transitive relationship (e.g. a cabbage, a hare and a fox with respect to the ”eat” relation), and a distractor object. They found that people would gaze at the cabbage upon hearing a sentence like (2-a) before actually hearing the second noun phrase, and would respectively gaze at the fox in sentences like (2-b). This means that people were anticipating the correct relationship between the hare and the eating event. One can therefore conclude that role assignment has been achieved at the point when the verb (*frisst* in example sentences (2)) is processed. If we assume that the syntactic relations have to be established before role assignment can be performed, the evidence from these experiments suggests full connectedness at the verb.

(2) a. Der Hase frisst gleich den Kohl.
The Hare-nom will eat soon the cabbage-acc.
b. Den Hasen frisst gleich der Fuchs.
The Hare-acc will eat soon the fox-nom.

Evidence from experiments on Japanese furthermore indicates that humans build compositional structures by connecting NPs in a grammatically constrained fashion in advance of encountering the verb, which is the head of the sentence and establishes the connection between the NPs (Aoshima et al., 2007).

Further evidence comes also from an English eye-tracking experiment (Sturt and Yoshida, 2008). In (3-c) the negative element c-commands and thus licenses use of the word *ever* later on in the sentence. This is not the case for sentences like (3-a), where processing difficulty can thus be expected at the point where the mismatch is detected. Reading times are indeed found to be longer for the word *ever* in condition (3-a). This indicates that the structural relations necessary for computing the scope of negation in sentences like (3) were available early during the processing of the relative clause, in particular before its verbal head had been processed. Thus, the modifiers *ever* or *never* must have been immediately incorporated into the structure.

(3) a. Tony doesn't believe it, but Vanity Fair is a film which I ever really want to see.
b. Tony doesn't believe it, but Vanity Fair is a film which I never really want to see.
c. Tony doesn't believe that Vanity Fair is a film which I ever really want to see.
d. Tony doesn't believe that Vanity Fair is a film which I never really want to see.

How easily the all words can be connected at each point in time under one root in practical parsing depends strongly on the grammar formalism used. We will review the ability of different established grammar formalisms to be used for incremental parsing in Section 6.3.

6.1.1.3 Experimental Results Challenging Strict Incrementality

While the above phenomena provide evidence for a strong degree of connectedness, there are also findings from other studies that suggest that sentence processing is not fully incremental, or at least that the valid prefix property, meaning that only analyses that are compatible with the interpretation so far are followed, is not always observed by humans. *Local coherence effects* are often interpreted as evidence that humans adopt a locally coherent interpretation of a parse, or experience interference effects by locally coherent structures which are however not compatible with the incremental interpretation.

Tabor et al. (2004) showed that participants are slower to read object-modifying reduced relative clauses (RCs) like the one shown in Example (4), when the RC verb is part-of-speech ambiguous than when the verbs POS tag is unambiguous.

(4) The coach smiled at the player tossed a frisbee by ...

In the example sentence, the word *tossed* could be a simple past form or a past participle form. If the ambiguous word *tossed* is replaced by a word that is unambiguously a past participle, such as *thrown*, the sentence becomes significantly less difficult. See Section 9.1.6 for a more detailed discussion of the study. This result is problematic for any fully incremental framework because the main-verb interpretation is incompatible with the global context and should thus be ignored by the processor, and hence not influence reading times. Local coherence effects have been successfully modelled using

a bottom-up CCG parser (Morgan et al., 2010) which does however not implement full connectedness.

A different account was proposed by Gibson (2006), who suggests that difficulty arises because the top-down global syntactic analysis conflicts with the bottom-up part-of-speech disambiguation, which are being done at the same time. These two analyses would compete and cause conflicts if they do not match. He suggested a formalisation of the difficulty as inversely proportional to context-independent probability of the POS-tag given the word multiplied by the smoothed syntactic expectations. This explanation does not necessarily require building up a representation for the locally coherent string, and could still be reconcilable with incremental processing (and the valid prefix property). Bicknell et al. (2008) tested this hypothesis on the Dundee Corpus, and found supportive evidence for processing difficulty arising from conflicting global interpretation vs. POS tag bottom-up analyses.

Other psycholinguistic effects that can be interpreted as evidence against full connectedness, and in favour of keeping unconnected structures around for later interpretation are reported in (Frazier and Clifton, 1996; Traxler, 2007; Swets et al., 2008). These studies suggest that modifiers like relative clauses are only attached when necessary. In order to account for these kinds of local coherence effects within an incremental framework, it seems necessary to assume that people sometimes remember the interpretation of a prefix imperfectly.

6.1.1.4 Discussion

The theory proposed in this work assumes full connectedness despite the indication of the existence of local coherence effects (which may find other explanations that may be compatible with incrementality, such as imperfect memory or interference from pre-syntactic processes). Connectedness provides a natural and elegant way of explaining why humans predict upcoming linguistic material (see Section 6.1.2 for a discussion of evidence of prediction): predictions are needed in order to maintain fully connected structures. Another challenge is the difficulty of arguing for a specific degree of connectedness, in particular since each study only makes claims about connectedness at a specific word in a specific construction. The theory proposed here therefore assumes the most simple and extreme interpretation of incrementality, full connectedness.

6.1.2 Prediction

"Prediction" here refers to the process of forming expectations about upcoming input. Evidence for prediction mainly refers to short-lived predictions affecting only the next couple of words. The plausibility of prediction in sentence comprehension has been discussed with respect to what the benefits of prediction would be, given that making predictions must also be related to some cognitive effort. This section first summarises the discussion in the literature of the relationship between prediction and integration, and then presents recent experimental evidence for prediction.

6.1.2.1 Prediction vs. Integration

Facilitatory effects have been observed for highly predictable words, which are read faster. However, for many of the experiments, it can also be argued that this facilitatory effect stems from easier integration of predictable words with the context.

People have therefore asked the question, whether a prediction process can be motivated – how would predictions help language comprehension? If they don't, why assume this additional mechanism? Some have argued that making predictions seems like a waste of a lot of processing effort, given that input is going to come up soon anyway: making predictions would unnecessarily take processing resources away from processing current input. Arguments that promiscuous prediction is computationally inefficient have for example been made by Charniak (1986); Singer et al. (1994); Jackendoff (2003); Marslen-Wilson and Welsh (1978).

On the other hand, several groups of researchers have argued for the existence of prediction in sentence processing, and have pointed out that benefits of generating predictions during comprehension are more rapid comprehension, and robust interpretation of ambiguous or noisy input (Pickering and Garrod, 2007). They argue that the language production system is used during comprehension to generate these predictions.

Evidence from word naming studies, where the task was to name a word that reflected the expected predictions (also referred to as forward inferences) of a short text, indicates that predictions are very short-lived, as facilitatory effects were only found directly after the sentence which was supposed to trigger the predictive inference (Keefe and McDaniel, 1993) but not when additional materials or a pause had intervened between the trigger sentence and the naming task (Potts et al., 1988; Singer and Ferreira, 1983). Recent evidence for predictive inferences comes from

Altmann and Kamide (2007) who found in a visual world experiment that participants looked to an empty glass on hearing “The man has drunk”, but to a full glass when hearing “The man will drink”. Further evidence for the existence of prediction comes also from the direct study of brain activations. Federmeier (2007) presents evidence for the existence of parallel predictive and integrative processing in language comprehension based on event-related potentials (ERP) tracked during language comprehension. Results indicate that top-down predictive processes are processed in the left brain hemisphere while bottom-up integration processes take place in the right hemisphere. The evidence for predictive processes stems from experiments showing that a strongly constraining (and hence strongly predictive) context leads to stronger N400 effects when the anticipated word is not encountered than weakly constraining contexts. Integration based accounts cannot account for the detected difference as integration ease was balanced based on pre-tests with Cloze probabilities, and integration-based theories would therefore predict the same integration difficulty in both contexts. Federmeier concludes that prediction seems important for language comprehension, but that it is also related to some processing cost and that the ability to make predictions deteriorates with age, showing that prediction effects are weaker for older adults.

6.1.2.2 Experimental Evidence for Prediction

Recent studies provided evidence that humans predict bits of the sentence that have not been preceived yet based on what they have heard so far, e.g. in studies by Kamide et al. (2003); Tanenhaus et al. (1995); van Berkum et al. (2005); Staub and Clifton (2006); Yoshida et al. (2009). The study by Kamide et al. (2003), see Examples (2) earlier in this chapter, provides not only evidence for connectedness, but also for prediction. Participants’ anticipatory eye-movements to the correct argument of the verb (as opposed to some other object on the screen) indicate that humans are predicting the upcoming argument, at least to the level of its semantic sort. Similar experiments have also been conducted for English, where the same effect was shown (Kamide et al., 2002).

Evidence for predicting a specific lexical item was found in an ERP experiment (van Berkum et al., 2005), where subjects heard Dutch stories that supported the prediction of a specific noun, see Example (5). To probe whether this noun was anticipated at a preceding indefinite article, stories continued with a gender-marked adjective whose suffix mismatched the upcoming noun’s syntactic gender. Adjectives that were inconsistent with the predicted noun elicited a differential ERP effect, which disappeared in a control condition where no specific noun could be predicted based on con-

text. Similarly, prediction-inconsistent adjectives slowed readers down before the noun in self-paced reading. These findings suggest that people can indeed predict upcoming words in fluent discourse, and, moreover, that these predicted words can immediately begin to participate in incremental parsing operations. Similar results were found for English in a study on the use of "a" vs. "an" as an indefinite determiner (Delong et al., 2005), who found larger N400 effects for indefinite articles that mismatched the expected upcoming noun (e.g., *The day was breezy, so the boy went outside to fly an...* where the word *kite* was most expected).

(5) *context:* De inbreker had geen enkele moeite de geheime familiekluis te vinden.
[The burglar had no trouble locating the secret family safe.]

a. *consistent:* Deze bevond zich natuurlijk achter een groot$_{neu}$ maar onopvallend schilderij$_{neu}$.
[Of course, it was situated behind a big$_{neu}$ but unobtrusive painting$_{neu}$.]

b. *inconsistent:* Deze bevond zich natuurlijk achter een grote$_{com}$ maar onopvallende boekenkast$_{com}$.
[Of course, it was situated behind a big$_{com}$ but unobtrusive bookcase$_{com}$.]

Another piece of evidence for prediction is the *either... or* construction. Results by Staub and Clifton (2006) show that hearing the word *either* triggers prediction of *or* and the second conjunct: reading times on these regions were shorter in the *either* condition, and participants also did not misanalyse disjunctions at sentence level as noun disjunctions in the condition where *either* was present. As an example, consider the sentence in (6). Here, the region *or an essay* is processed more quickly in (6-a) than in (6-b).

(6) 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.

As Cristea and Webber (1997) point out, there are a number of constructions with two parts where the first part can trigger prediction of the second part, similar to *either... or*. A related form of prediction is syntactic parallelism; experimental findings by Frazier et al. (2000) indicate that the second conjunct of a coordinate structure is processed faster if its internal structure is identical to that of the first conjunct. This can be seen as a form of prediction, i.e., the parser predicts the structure of the second conjunct as soon as it has processed the coordinator.

Very recently, Yoshida et al. (2009) argued that in a study of sluicing constructions in English, the parser predicts sluicing structure and uses the information to resolve anaphora binding. Examples for experimental items from the study are shown in (7).

(7) Nicole's father heard several stories during the holiday party, but it's not clear

a. which story of himself from the party her grandfather became so terribly upset over.
b. which story of herself from the party her grandmother became so terribly upset over.
c. over which story of himself from the party her grandfather became so terribly upset.
d. over which story of herself from the party her grandmother became so terribly upset.

At the point of *herself*, participants exhibited longer reading times in condition (7-b), because of the gender mismatch between *herself* and *Nicole's father*. This means that they must have predicted a sluicing construction at the point of processing *herself*, which in turn means that the structure up to *herself* must have been completely connected at that point. In a structurally-blind interpretation, or an account where the sluicing construction has not been built up and connected, the gender mismatch effect cannot be explained. Furthermore, they must have integrated the pronoun directly when it was processed, thus being able to resolve the anaphora binding. If the pronoun had not been integrated directly, we would only expect a later effect. In the control conditions (no gender mismatch as in Example (7-a), or no sluicing as in sentences (7-c) and (7-d)), reading times on the critical region were not significantly longer.

6.1.2.3 Verification

Whenever syntactic structure is predicted, we assume that it will be necessary to validate these predictions and to match up the predicted structure with what is actually encountered later on. The idea of verification bears similarities to integration in DLT, where arguments are integrated at the heads. Under the assumption of full connectedness, these heads are usually predicted earlier on. Experimental results on locality effects can be re-interpreted in terms of processing difficulty incurred through verification. This aspect is evaluated in Chapter 9.1 based on the full specification of the prediction theory and its implementation, (see Section 6.2 and Chapter 8).

6.1.2.4 Grain Size of Predictions

An important open problem in any theory of prediction is to identify the grain size of the predictions. Should a specific word, semantic sort, just the part of speech, or the syntactic structure be predicted? It may also be necessary to adapt the prediction level to the type of prediction cue seen. A related question is how far into the future predictions are made: just the next word, the next phrase, to the next possible end of sentence, even further?

One possibility would be to predict what's necessary in order to build a plausible and grammatical sentence under the full connectivity assumption. Such a notion of prediction would be conservative in that it predicts only what is minimally necessary to satisfy the assumptions of producing fully connected structures. Prediction necessitated by the connectedness assumption can happen e.g. in cases where two dependents but no head has been seen. The two dependents can only be connected into a single structure, if their common head (or at least a structure that requires a common head for them) is predicted. Similarly, when a grandparent and a child are seen, but not the parent, then the connecting structure between grandparent and child has to be predicted in order to achieve full connectivity.

Based on the experimental evidence outlined above (Kamide et al., 2003), predictions should also be generated based on a word's subcategorization frames. For predictions which are generated through subcategorization, the practical question arises of how to exactly define the subcategorization frames. The difference between arguments and modifiers seems to be gradual rather than categorial. Linguists have tried to differentiate between obligatory and optional constituents in language, such as in the X-bar theory, but it has been found that the distinction is notoriously difficult also for humans (e.g. in annotation, as can be seen in the disagreement between argument and modifier annotations from different resources such as PropBank (Palmer et al., 2003) vs. FrameNet (Johnson and Fillmore, 2000) and VerbNet (Kipper et al., 2000), as discussed e.g. in McConville and Dzikovska (2008)).

Another possibility is to always predict all possible structures (based on the grammar). Prediction grain size then mainly depends on the shape and independence assumptions of the grammar rules, and could potentially lead to making a very large number of very detailed predictions. The frequent generation of very detailed predictions however seems cognitively implausible due to the large prediction space this would create. Another question is the abstraction level for predictions. Should one

predict just the existence of an NP, or its internal structure, or its head? Clearly, more research is needed to learn more about the granularity of prediction that humans make.

6.1.2.5 Discussion

While it is still controversial whether explicit prediction takes place in human sentence processing, and whether it occurs on a regular basis during processing or just in very specific situations, we think that the evidence is well compatible with a prediction account. As discussed in the previous section, it is however difficult to pin down a specific level at which prediction happens based on the evidence so far – these results cover too few data points in the full space of where prediction could happen. This gap should be filled by conducting more experiments that can inform prediction grain size, and does not pose a fundamental problem to the concept of prediction.

6.1.3 Serial vs. parallel processing

Serial processing means that only one single analysis is processed at a time, usually entailing that the parser must have some kind of back-tracking mechanism so it can go back to an earlier point in the sentence and resume processing with an alternative analysis once it is clear that the current analysis is not viable. Serial processing is difficult to reconcile with some findings such as unforced reanalysis. Unforced reanalysis means that people can adopt an interpretation first which they then revise in favour of another analysis before having encountered a point in the sentence where the first analysis has become impossible or ungrammatical (Steedman and Altmann, 1989). In serial processing the processor cannot compare its current analysis to possible alternatives, and therefore is theoretically not able to give it up in favour of an alternative one. Instead, one would have to assume that rewrite-rules can get triggered, or that something causes the current analysis to be abandoned and a new one to be started. The notion of rewrite rules may be problematic because of the overhead of also having to rewrite role assignments and other semantic interpretations – this seems rather complex. On the other hand, it also seems difficult to pin down what causes a parse to be abandoned in unforced reanalysis cases. Given just the syntactic component, it is not clear how to decide on a threshold for when the parse is too bad and should be given up in the absence of material to normalise probabilities against. Instead, one would have to assume some conflict for example between the semantic interpretation and the syntactic analysis to trigger reanalysis (Frazier and Clifton, 1998).

Under parallel processing on the other hand, changing interpretations is straightforward: since several paths are followed at the same time, one can compare their probabilities / plausibilities and change to the more likely one at any given point. A complete parallel parser that follows all possible analyses at each point in time is however also implausible, given processing difficulty effects as seen in garden path sentences: a fully parallel model would have the correct interpretation available as well, and therefore should not lead to the difficulties observed in garden path sentences. Therefore, a conceptualisation as ranked parallel parsing has been suggested as an alternative to serial parsing. In a ranked parallel parser, analyses with very low probability or rank are discarded. In a garden path sentence, the correct analysis would have been discarded and reanalysis would be initiated when none of the currently maintained analyses are compatible with further input. The parser might then discard analyses less readily in a second run (under the assumption that the reader pays more attention, and hence allocates more resources to the parsing process, which would reflect in more memory allocation in the parser).

Discussion

People have found it notoriously difficult to come up with definite answer concerning serial or ranked parallel processing (Lewis, 2000; Gibson and Pearlmutter, 2000). We here assume ranked parallel processing. In addition to the memory restriction implemented through a finite beam of maximally maintained analyses, our theory models a limited memory also by restricting the number of predictions maximally maintained for each analysis (we also show in Section 8.2.3 that nothing beyond this limit is needed for parsing text like the Wall Street Journal).

6.1.4 Broad-coverage

Theories for syntactic processing are usually inspired by observations from very specific structures, such as garden path sentences, relative clauses, verb-final constructions, centre-embedding, ambiguous PP-attachment, idiom processing, case ambiguity, direct object vs. sentence complement ambiguity, etc., and often rather extreme versions of these structures were used to find reliable effects.

But in order for a theory to claim that it is a theory of syntactic processing in humans, it should not only be able to explain the pathologies in human processing, but also account for processing facilitation and behaviour on a wide variety of structures.

Theories should be evaluated on material that humans encounter in their daily lives and not exclusively on unnatural special cases, such as garden paths or difficult constructions that push the border of what humans can process. An important question to ask is therefore whether the existing theories scale up to naturally occurring, contextualised text, and whether syntactic structures have any measurable influence on such contextualised reading (Brants and Crocker, 2000).

The aim in this thesis is therefore to develop a theory of sentence processing that makes predictions for a wide variety of structures, instead of focusing on very specific sub-constructions.

6.2 Design of the Sentence Processing Theory

A main goal of this thesis is to construct a cognitively plausible model of human sentence processing, i.e. one that adheres to the specifications outlined in Section 6.1, such as incrementality, connectedness, making predictions (as far as humans do), and verifying them against upcoming events. Furthermore, we conceptualise sentence processing in a parallel framework and specify that the theory should be general enough to scale up to naturally occurring text, in order to account for difficulty incurred when processing broad-coverage text, as well as explain well-established experimental psycholinguistic findings.

Given these basic design decisions, we here define the link between the parsing process and processing difficulty. We thereby also take into account what we learnt from the broad-coverage analysis of previous theories, as discussed in Chapter 5: we observed that the forward-looking aspect of surprisal and the backward-looking aspect of DLT integration cost explained complementary bits of the processing difficulty found in the corpus. Furthermore, literature on these theories also found them to explain a different set of experimental findings. Therefore, our theory proposes to draw from both theories and unify them in a single concept of processing difficulty. We formulate processes in terms of cognitively plausible constructs, such as memory restrictions, activation and decay.

6.2.1 Processing Difficulty

We propose that processing difficulty be calculated incrementally as the sentence unfolds. Difficulty occurs through surprisal. If the perceived input is incompatible with

analyses that had a lot of probability mass and thus violates expectations, difficulty ensues.

Secondly, difficulty arises at integration time, when validating predicted structures against what is actually encountered. The amount of difficulty generated in verification depends on (a) how difficult the prediction was and (b) on how recently the prediction was made: if the prediction has decayed a lot, more difficulty arises than when a structure was predicted very recently. In our parsing model, we will therefore need to keep track of when a structure was predicted, and we do this through *time stamps*. There are lots of different ways people have proposed to quantify decay in previous work, e.g. counting distance in words, distance in terms of levels of embedding, number of intervening discourse referents, amount of memory interference by related items etc. Any of these accounts could be implemented within the sentence processing theory. As a first approach, we here use the simple measure of distance in words. A time stamp for a prediction is set to the number of the word at which the prediction was triggered. Predictions can also be reactivated if they are used in other operations such as substitution or adjunction.

Point (a), whether the verification of a prediction is more difficult when the original prediction was difficult, is also controversial: On the one hand, very frequently occurring predictions should be very easy to verify, as the parser does this very often and structure matching should be easy. On the other hand, one could argue that if a lot of resources are spent on a complex prediction, this should also be remembered better and hence easier to verify. This would point to using a variable decay factor, or more sophisticated memory retrieval model, e.g. as suggested in Lewis and Vasishth (2005).

Intuitively, the difference between a traditional surprisal account and the theory proposed here is that some of the probability mass (and hence the exact time when surprisal effects occur) can be shifted to different points during the processing because prediction trees are integrated before the full tree would otherwise be seen. This means that part of the cost of integrating the prediction tree into the structure is "paid" earlier through the forced commitment of connecting structures, while the rest of the probability mass (the size of the "rest" depends on prediction granularity, i.e. how completely a structure was predicted) is "paid" during verification. In addition to the nodes of the verification tree that were not predicted in advance, some difficulty is also incurred for remembering the predicted nodes, to the degree that their prediction has decayed. Note however, that despite these verification costs, previously predicted structures are generally easier to integrate than structures that were not predicted (in particular if they were

"completely" predicted all the way down to the lexical head and full subcategorization frame of the verification tree), because the surprisal of encountering a predicted word is very low.

Analyses can also be forgotten (pruned) from the set of analyses which are maintained in parallel if their probability falls out of the beam (as would happen for garden path sentences), or when they would require the processor to maintain a very complicated predicted structure that contains more nodes than allowed by a memory threshold (thus explaining centre embedding phenomena).

The model proposed in this thesis has two mechanisms that account for processing difficulty: the part that is related to surprisal quantifies the difficulty of the parser in terms of updating its representation of the analyses as the sentence unfolds. The verification process predicts difficulty based on a memory retrieval process for remembering and integrating newly encountered structure with previously predicted structure. These two types of processing difficulty thus model theoretically different aspects of human sentence processing.

A formalisation of this linking theory with respect to the PLTAG parser which we describe in Chapter 8, is spelled out in Section 8.7. We evaluate the sentence processing theory in Chapter 9.

6.3 Suitability of Grammar Formalisms

In the first part of this chapter, the underlying assumptions for the sentence processing theory: incrementality with full connectedness, parallel processing and prediction in combination with a verification mechanism, were motivated and outlined. In order to build a broad-coverage model that implements these assumptions, the parser, and hence a grammar formalism that the parser is based on, has to be able to accommodate these assumptions. While it is probably possible to tweak any grammar formalism such that it acts as if it was implementing the desired constraints, some formalisms might be inherently more suitable than others, in that the adaptation steps required are less difficult to realise. We here compare five grammar formalisms, Probabilistic Context Free Grammar (PCFGs), Combinatory Categorial Grammar (CCG), Tree-Adjoining Grammar (TAG), Dependency Grammar (DG) and Dynamic Syntax (DS).

Given the specifications of the sentence processing theory, the desirable properties of a grammar formalism to use for the implementation of this theory are incrementality and connectedness, as motivated in the first part of this chapter. Furthermore, it

is desirable that the formalism distinguishes between arguments and modifiers, such that subcategorised arguments can be predicted, but modifiers aren't. Another important point especially with respect to a final broad-coverage evaluation of the theory is tractability of parsing algorithms for the grammar formalism. A further criterion is easy linking of semantic interpretations to the syntactic structure, which is motivated by the incremental interpretation hypothesis and easy later extension to, and integration with, a semantic module. Finally, the generative power of the formalism can be used as a further argument for psycholinguistic suitability: a formalism that matches the generative power observed in human languages is inherently more plausible than one that over-generates (i.e. produces structures that are not observed in human languages) or one that under-generates, (i.e. cannot account for all of the phenomena encountered in human languages). Further psycholinguistic criteria are to match the degree of lexicalization in the human processor (even though evidence for this is controversial: There is both evidence for (Staub, 2007) and against (Mitchell, 1987) the immediate use of lexicalization information in verbs, and it's been argued that lexicalization may only come in at a later stage of the processing). Finally, the domain of locality is an aspect to take into account, which becomes particularly relevant for modelling the processing of idioms and non-compositional multi-word expressions, in particular in relation to the link to a semantic interpretation for such structures.

An overview of these criteria and how formalisms satisfy them, is shown in Table 6.1 at the end of this chapter. The remainder of this section will discuss each of the criteria for each of the grammar formalisms.

6.3.1 Probabilistic Context-Free Grammar

Probabilistic Context Free Grammars are most commonly used for natural language processing applications among the grammar formalisms discussed here. There are lots of resources and algorithms with well-known properties around, so that is of course an argument in favour of using a PCFG.

Incrementality / Connectedness Top-down parsing and left-corner parsing with left-transformed PCFG grammars allows to realise incrementality with full connectedness in a PCFG parser.

For psycholinguistic plausibility it is relevant to choose an arc-eager left-corner parser rather than an arc-standard left-corner parser, as the arc-standard left-corner

parser would lead to asymmetric predictions for the difficulty of left- vs. right-branching structures, while arc-eager left-corner parsing would only predict centre embedding to be more difficult and hence model psycholinguistic evidence better (Thompson et al., 1991; Resnik, 1992).

Argument vs. Modifier Distinction A PCFG can satisfy this criterion if a suitable grammar is chosen. For example, rules including modifiers should be binarised to express that the *ADJ* is optional within the *NP*. So a rule like

$$\text{NP} \rightarrow \text{DT ADJ N}$$

should be replaced by a set of rules like the following:

$$\text{NP} \rightarrow \text{DT NP'}$$
$$\text{NP'} \rightarrow \text{ADJ NP'}$$
$$\text{NP'} \rightarrow \text{N}$$

while it is necessary that a verb keeps all the required entities in the same rule:

$$\text{VP} \rightarrow \text{V OBJ OBJ2}$$

Furthermore, rules would need to be lexicalized – a formalism like TAG or CCG seems to do this more naturally.

Tractability A top-down fully connected incremental parser has been implemented (Roark, 2001b), both as a generative model (Roark, 2001a) and as a discriminative parsing algorithm (Collins and Roark, 2004). The discriminative model is however not suitable for estimating prefix probabilities (which is needed e.g. for calculating surprisal).

Semantic Interpretation A semantic interface could be designed that uses the incremental output of Roark's parser and links it to a semantic representation.

Lexicalization Not fully lexicalized in the sense of CCG or TAG, but can use features for stronger lexicalization.

Domain of Locality Very local, no long dependencies or larger tree structures.

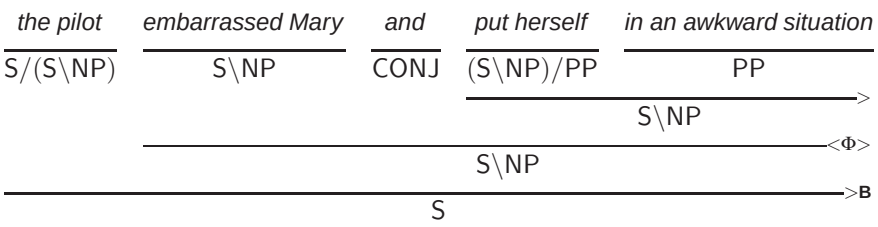


Figure 6.1: Binding would only occur after full processing of second conjunct according to CCG derivation. However, the empirical finding is that humans experience difficulty of gender mismatch as soon as they hit the reflexive pronoun.

Generative Power Only context-free, so less powerful than the human processor has been argued to be based on languages that contain context-sensitive constructions.

6.3.2 Combinatory Categorial Grammar

Combinatory Categorial Grammar (CCG) is a linguistically more expressive grammar formalism (Steedman, 2000). In CCG, the language-specific knowledge about the grammar is stored in the lexicon. There is then a finite set of rules that allow the lexical categories to combine. A detailed discussion of CCG and how it could be used for incremental, fully connected parsing is available in Appendix A.

Incrementality / Connectedness CCG was originally designed as an incremental formalism, and it is often claimed that CCG supports fully incremental derivations because of its very flexible notion of constituents. Besides a normal form derivation (which is the derivation that uses least rules), non-standard constituents can be combined via the application of composition. CCG supports all phrases as constituents that are licensed by the grammar, and Steedman (2000) claims that the constructions supported by CCG are the ones that can be shown to be interpreted incrementally by humans.

However, there is also evidence for cases where connectedness in sentence processing is stronger in humans than under a CCG derivation. We briefly discussed the incrementality study by Sturt and Lombardo (2005) in experimental items (1) in Section 6.1.1. Sturt and Lombardo's (2005) experiment shows an example of where standard CCG is not incremental enough to explain empirical findings (see Figure 6.1), because it would only construct the syntactic connection between *herself* and *pilot* at the end of the sentence.

Other examples of constructions where CCG is not incremental enough for fully

connected parsing include object relative clauses, see Figure 6.2. Hence CCG would have to be modified, e.g. by changing some of the categories, in order to make it suitable for strictly incremental parsing, especially for the object relative clause which is one of the important evaluation cases in this work. It was not obvious how to do that without changing the generative power of CCG and causing over-generation (see Appendix A for more detail).

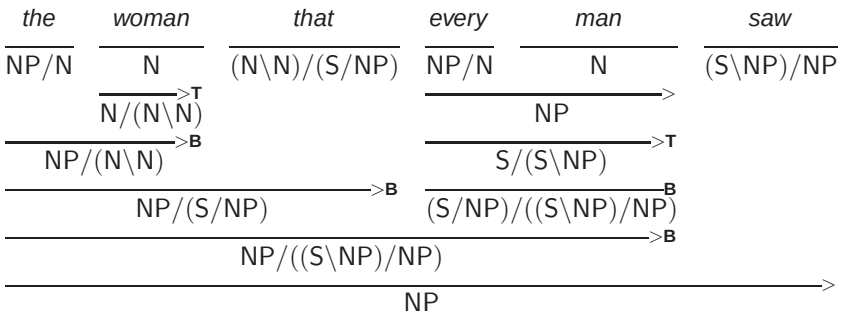


Figure 6.2: Example of incrementalized derivation for object relative clause in CCG. It is not possible to make a fully incremental version inside the ORC NP “every man”.

Generative Power CCG is mildly context-sensitive and hence more powerful than PCFGs. It can explain many linguistic phenomena, e.g. Dutch serial dependencies. CCG can also capture long distance dependencies better than CFGs.

Tractability Tractability for CCG depends on whether the unary operations type-raising and geaching are lexicalized or not. Best case tractability is $O(n^6)$. However, implementations of CCG, in particular the C&C parser (Clark and Curran, 2004) are very fast (it uses a discriminative model though). Hockenmaier and Steedman’s (2002) parser uses a generative model, and also achieves respectable accuracy and speed.

Semantic Interpretation CCG has a direct interface to semantic interpretations, even though the semantic interpretations are slightly non-standard. Baldridge and Kruijff (2002) suggest annotating CCG lexical categories with a modality, which indicates dependencies and would be stable against type-raising. Clark and Curran (2007) describe how to convert CCG dependencies into grammatical relationships in Depbank style. The conversion requires some amount of hand-written rules to transform the CCG dependencies into Depbank dependencies, as well as changing some manual annotations of the dependencies in the CCG lexicon and post-processing for matching templates.

An incremental CCG derivation can directly account for partial semantic interpretations of the sentence at each point. It should not be a problem to get from the CCG semantics to semantic roles.

Arguments vs. Modifiers Arguments and modifiers are distinguished in CCG, one can identify modifiers by the fact that they yield the same category that they take (which corresponds to auxiliary trees in TAG). In CCGBank Hockenmaier and Steedman (2007), heuristics were used to distinguish arguments from adjuncts, but new annotation in particular for NPs has been added since (Honnibal and Curran, 2007; Honnibal et al., 2010).

6.3.3 Tree Adjoining Grammar

Tree-adjoining grammar (TAG) was developed by Joshi et al. (1975) as a linguistically inspired grammar formalism. While CCG and PCFGs are string-rewriting formalisms, TAG is a tree-rewriting formalism. Like CCG, TAG stores all knowledge about the grammar in the lexicon, which contains tree structures. Tree structures for words can be linked together to form a sentence using two different operations, substitution and adjunction.

There exist a number of different versions of TAG, which are referred to in the thesis. The most important ones are *Lexicalized Tree Adjoining Grammar* (LTAG), where each tree in the lexicon must have at least one lexical anchor. LTAG grammars for a number of different languages have been created – the biggest of them is the XTAG effort for English (The XTAG Research Group, 2001). *LTAG-spinal* (Shen and Joshi, 2005), where all LTAG trees only have "spines" (i.e. the path from the lexical anchor to the root of the tree) but no substitution or foot nodes. An approach to defining a version of TAG which allows for full connectedness is *Dynamic Version of TAG* (DV-TAG; Mazzei et al., 2007). Finally, a related but less powerful version of TAG is *Tree Insertion Grammar* (TIG) (Schabes and Waters, 1995), which also has the two basic operations of substitution and adjunction but is sufficiently restricted to only derive context-free languages. LTIG (Lexicalized TIG) trees are a subset of LTAG trees, excluding all those LTAG trees where the foot node in an auxiliary tree is not the leftmost or rightmost child in the tree.

Incrementality / Connectedness Standard TAG or LTAG do not allow for incremental fully connected processing. However, the Dynamic Version of TAG constitutes

an incremental, fully connected version of TAG. The problems encountered in CCG, concerning lack of connectedness for ORC relative clauses and coordination do not occur in DVTAG, see for example Figure 6.3.

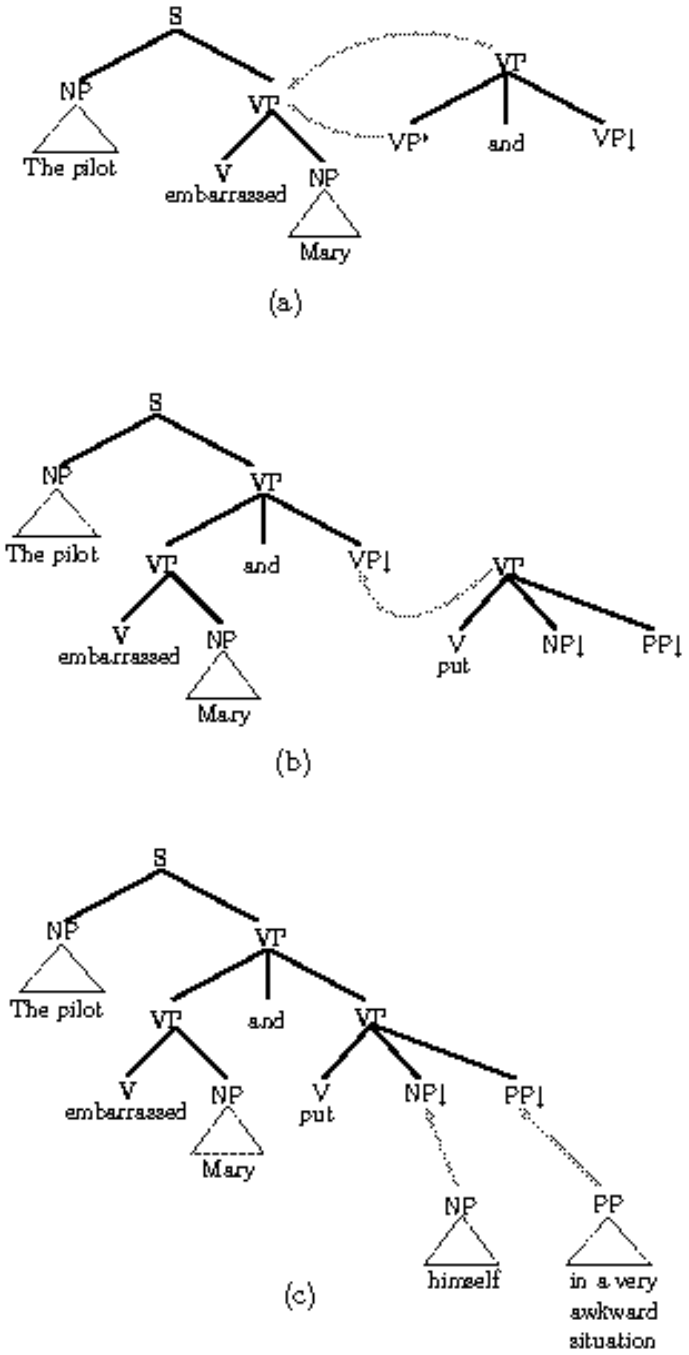


Figure 6.3: TAG derivation for Sturt et al.'s experimental sentence; graph taken from Mazzei (2005).

Generative Power The generative power of LTAG has been argued to be weakly equivalent to the generative power of CCG, and is stronger than the generative power of CFGs (Vijay-Shanker and Weir, 1994).

Extended Domain of Locality Another asset of TAG is its extended domain of locality. This means that TAG can e.g. capture the exact subcategorization frame of a verb in one rule (the verb's elementary tree) instead of using several syntactic rules $S \rightarrow NP\ VP$, $VP \rightarrow V\ NP$, or has to lose the VP node in a ternary rule $S \rightarrow NP\ V\ NP$, as one would have to do in a PCFG. Furthermore, a tree in the lexicon can have two or more lexical anchors, thus encoding idioms like "kick the bucket" as one entity, together with its semantic interpretation.

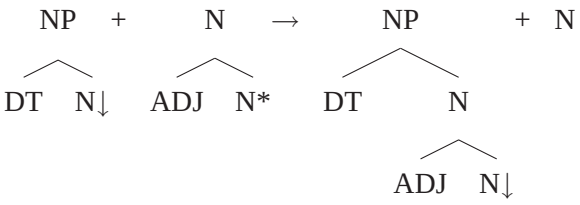
Tractability LTAG is parsable in $O(n^6)$, just like CCG. However, maintaining the valid prefix property (VPP) requires $O(n^9)$ processing time, see Joshi and Schabes (1997) for a more detailed discussion. In practice, TAG parsers (and their context-free cousins TIG parsers, which can parse in $O(n^3)$) achieve good parsing accuracy and speed. There is already an incremental LTAG parser available (Shen and Joshi, 2005), which is based on spinal LTAG. The performance is about 90% f-score for dependencies. However, it does not construct fully connected structures. Unfortunately, no parser for DVTAG, the strictly incremental version of TAG has been implemented, due to a large lexicon. There also exists a LTIG parser implemented as a generative model (a generative model is necessary for calculating prefix probabilities) (Chiang, 2000).

Arguments and Modifiers The LTAG bank was converted from the Penn Tree Bank and contains added information from PropBank, so it can be assumed to be similar to the amount of knowledge to the (heuristically disambiguated) CCGBank.

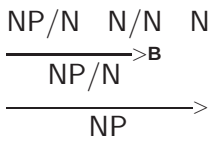
Semantic Interpretation The dependency structure which can be used to calculate the semantic interpretation is directly available through the derivation tree. For more detailed discussions, see (Mazzei et al., 2007; Kuhlmann, 2007).

Psycholinguistic Plausibility Lexicalized Tree Adjoining Grammar has been argued to be psycholinguistically plausible regarding aspects of language acquisition (the substitution operation is learnt before the adjunction operation), and disfluencies in language production (Ferreira, 2005).

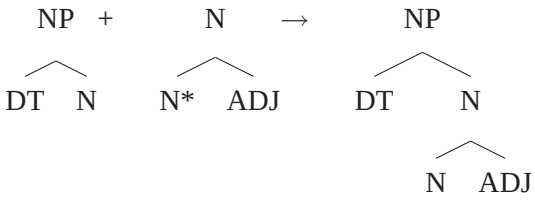
Further notes: Pre- vs. post-modification There is an asymmetry between pre- and post-modification in CCG in terms of the operations needed, which does not occur in TAG. To our knowledge, whether pre-modification and post-modification are computationally equivalent in human sentence processing is still an empirical question – some insight might be gained from French, which has both pre- and post-nominal adjectives[1]. The reason for the asymmetry is that for pre-modification, e.g. an adjective before noun, there is no type-raising necessary in incremental processing. On the other hand, for post-modification it is necessary to perform an additional type-raising operation (or to introduce an ambiguity in the supertag of the noun phrase that is being modified, if type-raising is lexicalized), see Figure 6.4(b) and (d). CCG uses one more operation for post-modification than it does for pre-modification, while TAG uses the same amount of operations, see Figure 6.4(a) and (c). Whether this difference causes processing difficulty predictions to be different between the pre- and post-modification depends on the linking theory.



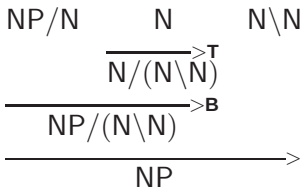
(a) TAG pre-modification



(b) CCG pre-modification



(c) TAG post-modification



(d) CCG post-modification

Figure 6.4: Comparison of pre- and post-modification in TAG and CCG

[1]However, these are not semantically and distributionally equivalent, therefore some other language where the pre- vs. post-modification position can be varied more freely might provide better evidence.

6.3.4 Dependency Grammar

Dependency grammar originates from the work of Tesnière (1959). Dependency grammars are by definition lexicalized, since only words can be nodes in the tree. This may also make the parsing task easier because no new nodes have to be postulated. The "missing" phrase structure also means that the formalism is less expressive, i.e. it is underspecified whether a modifier modifies the whole phrase or just the head of a phrase.

Incrementality / Connectedness Incremental deterministic dependency parsers like the MALT parser (Nivre, 2004) have received a lot of attention in recent years. However, the MALT parser uses a stack and does not support full connectedness. In very recent work, Menzel (2009) proposed a fully connected incremental dependency parser within the framework of Weighted Constraint Dependency Grammars. The difficulty with full connectedness in dependency parsing is that there are no non-terminal nodes. Therefore, if e.g. the head of two dependents has not yet been seen, some empty node must be predicted for these two nodes to depend on.

Argument vs. Modifier Distinction Labelled dependency arcs may specify the argument / modifier status of a dependent.

Semantic Interpretation Ease of semantic interpretation depends on whether the connections are labelled with their functionalities. If they are, semantic interpretation is straightforward.

Tractability and Implementation In practice, deterministic dependency parsers like Nivre (2004) have been shown to be fast and achieve competitive accuracy. An incremental (not fully connected) Nivre-style dependency parser with a small beam has been implemented by Marisa Ferrara Boston (Boston et al., 2008). Surprisal is calculated based on this parser.

Generative Power Mildly context-sensitive languages are described through crossing dependencies (Kuhlmann and Mohl, 2007).

6.3.5 Dynamic Syntax

Dynamic Syntax is a grammar formalism designed to directly reflect the left-right (time-linear) sequence of natural language processing (Kempson et al., 2005, 2001).

Incrementality / Connectedness The formalisms is conceptualised as an incremental formalism, and is fully connected, i.e. each word is integrated into the structure directly, there is no stack.

Arguments and Modifiers Dynamic Syntax differentiates between arguments and modifiers, arguments are predicted (their head introduces a "requirement" of having them) while modifiers are not.

Semantic Interpretation DS builds up a propositional structure instead of a standard syntactic tree, so it directly reflects the predicate-argument structure of the sentence.

Tractability and Implementation There exists a Prolog implementation of a Dynamic Syntax parser and generator by Matt Purver. The parser is word-by-word incremental (Purver and Kempson, 2004). The implemented lexicon however is tiny and there seem to be serious coverage and tractability issues. It would likely be much harder to obtain a version of the Penn Treebank to train a Dynamic Syntax parser on than for any of the other grammar formalisms, for which such converted treebanks are already available.

Generative Power The DS derived trees are characterisable in context free terms (since they are only functor/argument binary trees) but the system as a whole is characterised as context-sensitive in a general sense. However, formal characteristics seem to be unknown[2].

Domain of Locality Similar to TAG. Lexical packages can include several lexemes, and the actions can construct or annotate arbitrary tree structure.

[2] Comment based on personal communication with Ronnie Cann.

6.3.6 Discussion

Table 6.1 summarises the suitability results according to the important requirements for the suggested sentence processing theory. All of the grammar formalisms[3] can be processed incrementally, however, fully connected processing has only been shown to be possible for PCFGs (Roark, 2001a), in DVTAG (Mazzei, 2005), DG (Menzel, 2009) and DS (Kempson et al., 2001), although for DVTAG and DS no implementation that would scale up to broad-coverage processing is available, and for dependency grammars, no fully connected parsing procedure was available at the time when I considered this question for this work. All of the grammar formalisms support the distinction of arguments and modifiers, and a link to a semantic representation can also be established for all grammar formalisms, but in CCG and DS it is an integral part of the formalism. Psycholinguistic plausibility has been claimed for most strongly for Tree-Adjoining Grammar, see comment in Section 6.3.3. TAG is furthermore the only formalism beside DS to support a larger domain of locality.

criterion	PCFG	CCG	TAG	DG	DS
incrementality	+	+	+	+	+
full connectedness	+	–	+	+	+
arg / mod distinct	+	+	+	+	+
tractability	+	+	+	+	–
link to semantics	(+)	+	(+)	(+)	+
generative power	–	+	+	+	?
domain of locality	–	–	+	–	+
lexicalization	–	+	+	NA	–

Table 6.1: An overview of selection criteria by grammar formalism.

Taken together, the criteria seem to be best fulfilled by an adapted version of TAG, possibly similar to DVTAG. We decided against PCFGs due to their smaller generative power and small domain of locality. CCG was ruled out due to the incrementality problems outlined above. At the time, dependency grammar was mainly ruled out for lack of a fully connected parsing strategy, but also because of the small domain of

[3]The above list of grammar formalisms is of course not an exhaustive list of all existing grammar formalisms. Other incremental formalisms include Left Associative Grammar (LAG) (Hausser, 1986), and the a proposal for incremental structure building by Phillips (2003). Further established grammar formalisms include HPSG (Pollard and Sag, 1994) and LFG (Bresnan, 2001).

locality. Finally, Dynamic Syntax was ruled out based on tractability problems. The only implementations for DS are in Prolog and operate on toy language fragments.

6.4 Conclusions

In this chapter, we have motivated the properties that our theory of sentence processing should implement: incrementality, full connectedness, explicit prediction in combination with a verification mechanism and parallel processing. In addition, the theory should be specified and implemented such that an application to broad-coverage text is possible.

We then explained the mechanisms of the proposed sentence processing theory and outlined how processing difficulty is incurred. A more formal definition will be given in Section 8.7.

The last part of the chapter reviewed alternative grammar formalisms with respect to how well they conform to the specifications set out in our sentence processing theory, and argued that Tree-Adjoining Grammar (TAG) would be most suitable. Challenges posed by the choice of TAG are that a fully connected parser for an incremental version of TAG did not exist, and that the existing incremental version of TAG, DVTAG, requires further conceptual modifications to achieve greater psycholinguistic plausibility, in particular with respect to prediction grain size. These issues will be addressed in Chapter 7, which discusses a psycholinguistically motivated version of TAG (PLTAG), and Chapter 8, which describes the implementation and evaluation of an incremental fully connected predictive parser for PLTAG. Finally, the sentence processing theory suggested in this Chapter will be evaluated in its incarnation based on the incremental PLTAG parser in Chapter 9.

Chapter 7

PLTAG: A psycholinguistically motivated version of TAG

The last chapter outlined a new theory of sentence processing which assumes strictly incremental processing and contains an explicit mechanism for prediction and verification. An implementation of this theory must be based on a grammar formalism and parser that also adhere to the theory's assumptions. In this chapter, we describe a specially developed grammar formalism, PLTAG, which is a strictly incremental variant of Tree-Adjoining Grammar (TAG).

In the first part of the chapter, we motivate and define the new grammar formalism, and compare it to standard TAG. In Section 7.3, more detailed design questions concerning predicted entities and prediction granularity, also in relation with the sentence processing theory described in the previous chapter, are discussed.

Parts of the material presented in this chapter have been published as Demberg and Keller (2008b) at the TAG+9 workshop.

7.1 Limitations of Standard LTAG

7.1.1 An Introduction to LTAG

Tree-adjoining grammar (Joshi et al., 1975) is a tree-rewriting formalism. TAG stores all knowledge about the grammar as little tree structures, called *elementary trees*, see the trees in Figure 7.1(a) - (c) as an example. Here, we will only talk about Lexicalized TAG (LTAG), where all elementary trees have at least one *lexical anchor*, i.e. at least one leaf is a lexical item. There are two types of elementary trees: *initial trees* (see

Figure 7.1(a) and (c)), and *auxiliary trees* (see Figure 7.1(b)). Auxiliary trees are used to for a language's recursive structures, and are different from initial trees in that they contain exactly one *foot* node (marked with the * symbol). A foot node always has the same category as the auxiliary tree's root node. Both auxiliary and initial trees can have zero or more *substitution nodes* (marked with the ↓ symbol).

Elementary trees can be linked together to form the syntactic structure of a sentence (see Figure 7.1(d)) using two different operations, *substitution* and *adjunction*.

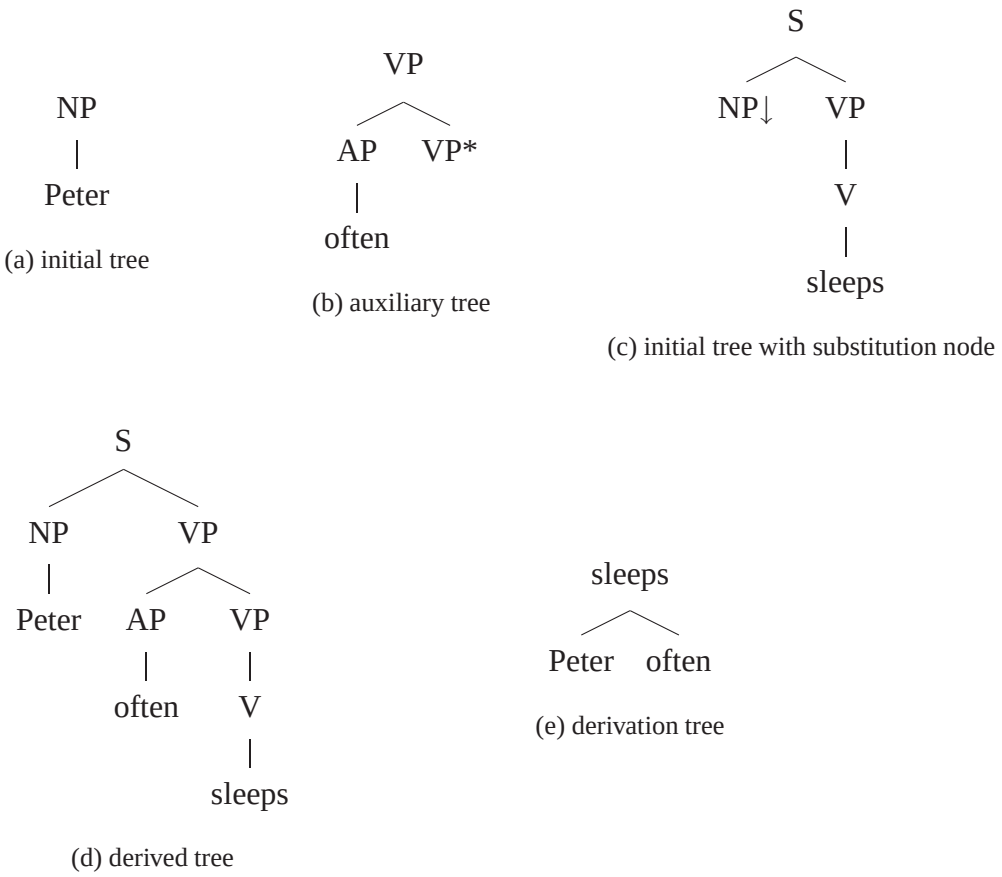


Figure 7.1: Examples for TAG elementary trees.

The substitution operation integrates an initial tree with a substitution site of the same category. For example, the initial tree for *Peter* in Figure 7.1(a) can be substituted into the substitution node in Figure 7.1(c). The adjunction operation can be thought of as two substitution operations: an internal node with a category matching the auxiliary tree's root node category is selected. The tree is cut apart at that point and the auxiliary tree is substituted in at this node. Then the lower bit of the tree is substituted into the auxiliary tree's foot node.

The derivation tree (see Figure 7.1(e)) encodes how the elementary trees were integrated with each other to construct the derived tree.

7.1.2 LTAG and Incrementality

Using standard LTAG with a lexicon of the typical linguistically motivated tree structures (like shown in the top row of Figure 7.2), it is not possible to derive even simple sentences such as *Peter often reads a book* incrementally. According to LTAG derivation rules, a derivation always starts with a tree whose root category is S. Therefore, the derivation can only start with the first word of the sentence if it happens to be a sentential head, which is not the case for most English sentences. Even if this rule about starting with an S-rooted tree was relaxed, and operations adjusted accordingly, the words *often* and *a* would still be out of order in our example sentence, see the derivation in Figure 7.2.

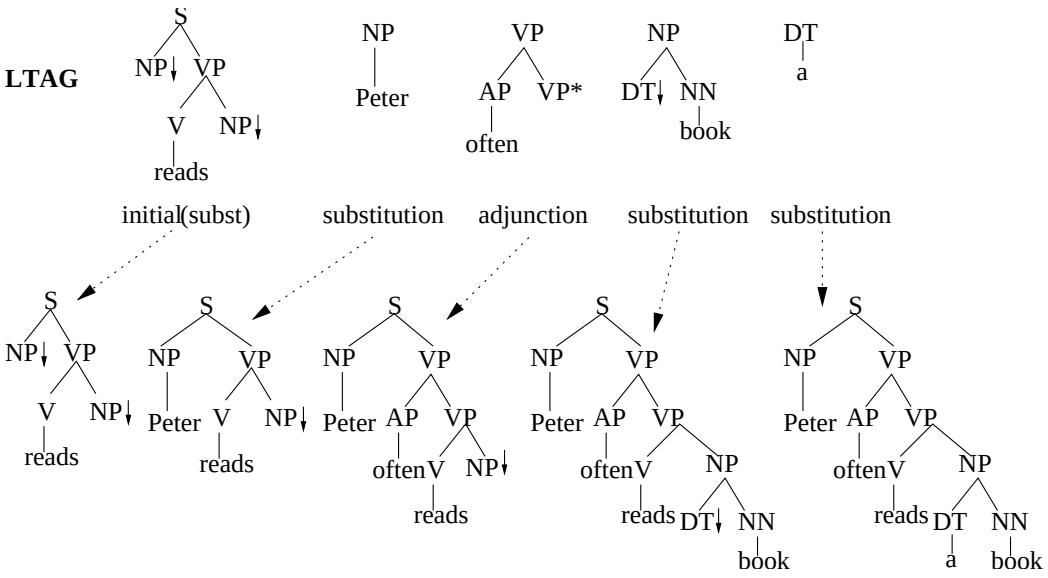


Figure 7.2: The most incremental derivation possible using LTAG for the sentence *Peter often reads a book*.

The next section proposes a new version of TAG, called Psycholinguistically motivated LTAG (PLTAG), which can overcome this limitation, by providing predictive structures – *often* can be integrated with *Peter*, because the missing structure of their common head, a verb, is predicted.

7.2 The PLTAG Formalism

PLTAG extends normal LTAG in that it specifies a lexicon of so-called *prediction trees* in addition to the canonical lexicon (which contains lexicalized initial and auxiliary trees). The role of the prediction trees is to provide the structure needed for connectedness, i.e. predict structure that is otherwise part of later trees, hence the name "prediction tree". The canonical lexicon is very similar to other LTAG lexica. Cases where PLTAG analyses differ from XTAG (The XTAG Research Group, 2001) analyses are discussed in Section 7.3.

The prediction lexicon consists of elementary trees which are usually unlexicalized and where each node has a special marker indicating that the node is predicted. The markers on the nodes consist of a superscript and / or a subscript, which indicate its predictive status. The super- and subscripts are used similar to the features in Feature structure based TAG (FTAG, see Vijay-Shanker and Joshi, 1988). A root node only has a subscript, while substitution and adjunction nodes have only superscripts. Inner nodes have both subscripts and superscripts. The reason for root, foot and substitution nodes only having half of the indices is that these nodes still need to combine with another tree in order to constitute a complete node. For example, if an initial tree substitutes into a substitution node, the node where they are integrated becomes a complete node, with the upper half contributed by the substitution node and the lower half contributed by the root node of the substituted tree. Similarly, in adjunction, the node where the adjoining operation is going to take place is broken up into its upper and lower half. The upper half combines with the auxiliary tree root node into a complete node, while the lower half combines with the auxiliary tree foot node into a new node. A fully indexed tree is shown in Figure 7.3. Note that unlike in Figure 7.3, nodes of canonical trees, or of complete derived trees are not annotated with indices in PLTAG. The indices are only used to mark predicted nodes, and are removed as soon as a node is verified.

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, PLTAG has a verification operation, which is needed to validate the nodes previously introduced by the integration of a prediction tree.

Verification is an operation that removes prediction indices from the prefix tree for all nodes that it validates, and can introduce additional nodes below the last node on

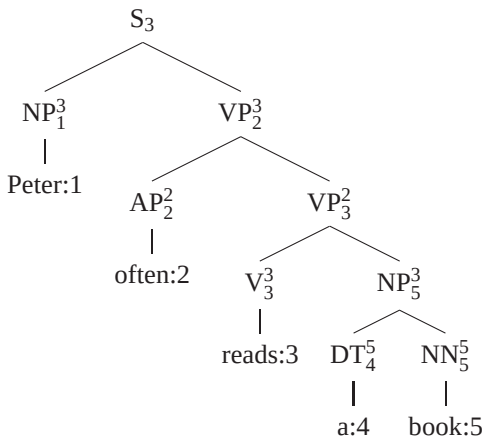


Figure 7.3: The indices at the tree nodes indicate which word each node comes from originally (words are numbered 1 to 5).

the spine[1] of the prediction tree or to the right of the spine (this restriction to the right side of the spine reflects the asymmetry of incrementality). The elementary tree used in a verification operation (also referred to as the *verification tree*) must be a canonical tree, and must match (we'll define this later in more detail) all predicted nodes with the same index, and no other ones. In brief, a verification tree matches the structure of a prediction tree if the two trees have all nodes in the same order, with the exception that the verification tree may contain additional nodes at the bottom of the spine or to the right side of the spine.

Figure 7.4 provides examples for each of the three operations. The operations are discussed in detail in the context of the PLTAG parser in Section 8.3.3.

A valid PLTAG derived tree for a sentence is a tree structure which must not contain any nodes that are still annotated as being predictive – all of them have to have been validated through verification once the input string has been fully processed. As in other versions of TAG, the derived tree for a sentence may not contain any open substitution or foot nodes.

Psycholinguistically motivated Tree Adjoining Grammar (PLTAG) can thus be defined as a tuple $G = (S, N, T, I, A, PI, PA, F)$:

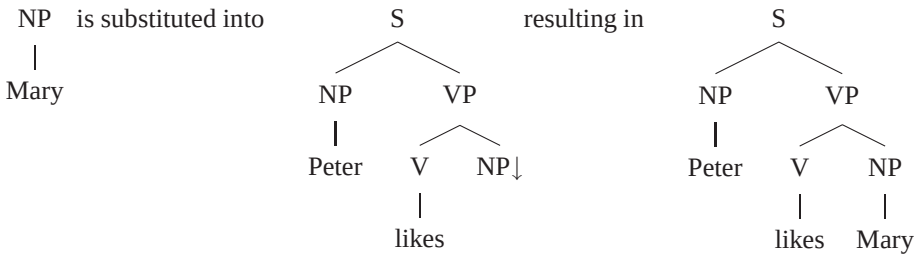
S: the non-terminal symbol that is the root of a derived tree

N: the set of non-terminal symbols

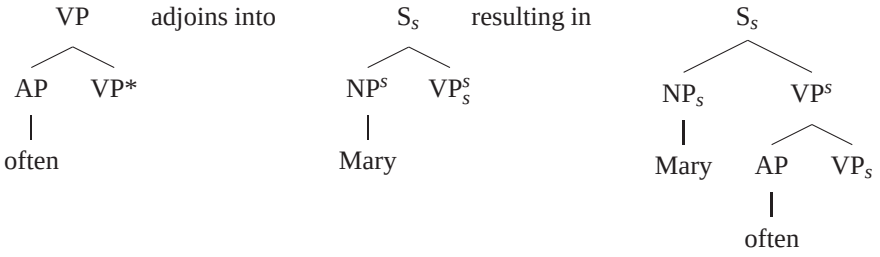
T: the set of terminal symbols

I: a finite set of initial trees

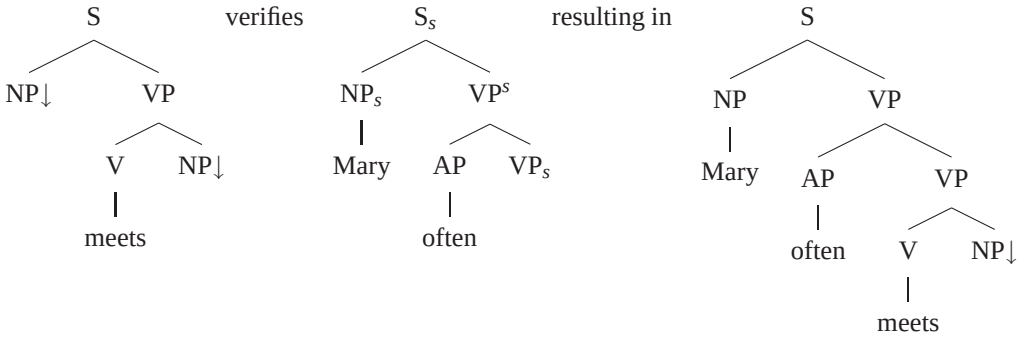
[1] The spine of a tree is the path from the root to its anchor leaf, this usually coincides with the head of that tree.



(a) substitution



(b) adjunction



(c) verification

Figure 7.4: Examples of PLTAG operations.

A: a finite set of auxiliary trees
PI: a finite set of initial prediction trees
PA: a finite set of auxiliary prediction trees
F: a set of indices that mark the non-terminals on prediction trees

All trees that can be generated by a PLTAG are composed of trees from *I*, *A*, *PI* and *PA*, through integration using the operations *adjunction*, *substitution* and *verification*, starting with the first word of the sentence and proceeding incrementally (see Section 7.2.1 for a definition of a PLTAG derivation). The language generated by a PLTAG is

the set of all terminal strings on the frontier of the trees that can be generated by the grammar.

7.2.1 Derivations in PLTAG

LTAG derivations are defined as starting with an initial tree whose root node is an S-node, and then applying the standard substitution and adjunction operations such that elementary trees are always integrated into the partially derived tree. An LTAG derivation is complete when every leaf node of the derived tree is labelled with a terminal symbol.

A *PLTAG derivation*, on the other hand, starts with the tree of the first input word, and then applies substitution and adjunction operations. In PLTAG, the new elementary tree can either be substituted or adjoined into the partially derived tree, or the partially derived tree can be substituted or adjoined into the elementary tree. If substitution or adjunction is applied to a *prediction tree* the nodes annotated with prediction markers in the resulting tree will have to be validated using the verification operation. This means that for each integration of a prediction tree, there has to be a verification operation later on.

A *partial derivation* for words $w_1..w_i$ in PLTAG contains only lexicalized leaves to the left of the rightmost lexical anchor. I.e. it must not contain any leaf nodes with prediction markers or open substitution nodes before word w_i (see Figures 7.5 and 7.6). If the partial derived tree is an auxiliary tree, its foot must be to the right of the lexical anchor. A PLTAG derivation is complete when every leaf node is labelled with a terminal symbol, none of the nodes in the tree is marked as predictive, and the root symbol of the derived tree is S.

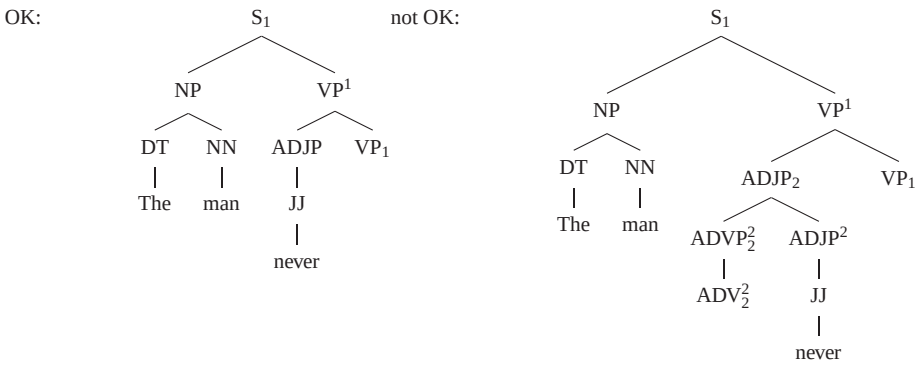


Figure 7.5: Example of a tree with an unverified prediction nodes.

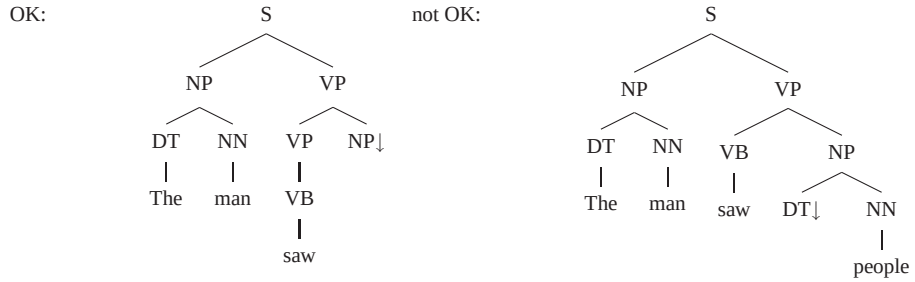


Figure 7.6: Example of a tree with open substitution nodes.

In order to better illustrate the intuitive relationship between LTAG and PLTAG, we compare the derivations of the sentence *Peter often reads a book* in LTAG vs. PLTAG. As we have seen earlier, the most incremental LTAG derivation we can generate has several words in the wrong order (see Figure 7.2). The words *read* and *book* are not in incremental order, because *often* and *a* are only inserted later. The PLTAG derivation, on the other hand, integrates all trees in correct incremental order, but makes use of prediction trees (marked with the indices ^{s} and $_{s}$), see Figure 7.7. The derivation starts with the initial tree for *Peter* and then substitutes it into a prediction tree. This means PLTAG would predict that *Peter* is the subject of a verb phrase. The auxiliary tree for *often* can then be adjoined into the predicted VP node. Next, the prediction can be validated: in fact it is compatible with the upcoming verb *reads*. Given that the verb is transitive, it subcategorizes for an NP, and we predict that in fact a determiner and noun might be coming up next, by substituting a prediction tree into the NP substitution site. The determiner *a* is then substituted into the DT prediction node, and finally the predicted noun *book* is encountered and the prediction verified.

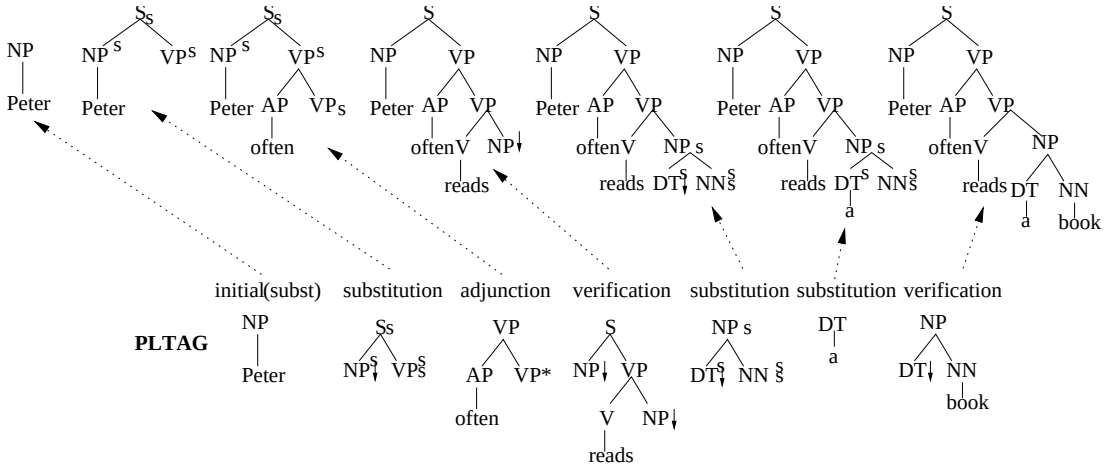


Figure 7.7: The derivation for the sentence *Peter often reads a book* with PLTAG.

The correspondence between an LTAG and a PLTAG derivation is intuitively straightforward. Let's go from the LTAG derivation to the PLTAG derivation (see Figure 7.8): For each misplaced (wrt. the incremental order) tree, we have to use a prediction tree instead, which matches the shape of the original tree, such that the original tree can later verify the prediction tree. The "misplaced" tree can then be moved to the correct incremental position for its lexical anchor, and will be integrated into the derivation using the verification operation. Hence, there are the same number and shapes of canonical trees in both derivations, and they are joined together using the same operations, except that in a PLTAG derivation a prediction tree and a corresponding verification operation is added for each out-of-order item. The function of the verification operation is to replace each prediction tree by its verification tree, such that in the final derived tree doesn't contain any prediction trees or parts of them.

Similarly, we can convert any PLTAG derivation into an LTAG derivation by replacing each prediction tree by the canonical tree that verifies it. The final derived trees for LTAG and PLTAG are identical (compare final derived trees in Figures 7.2 and 7.7).

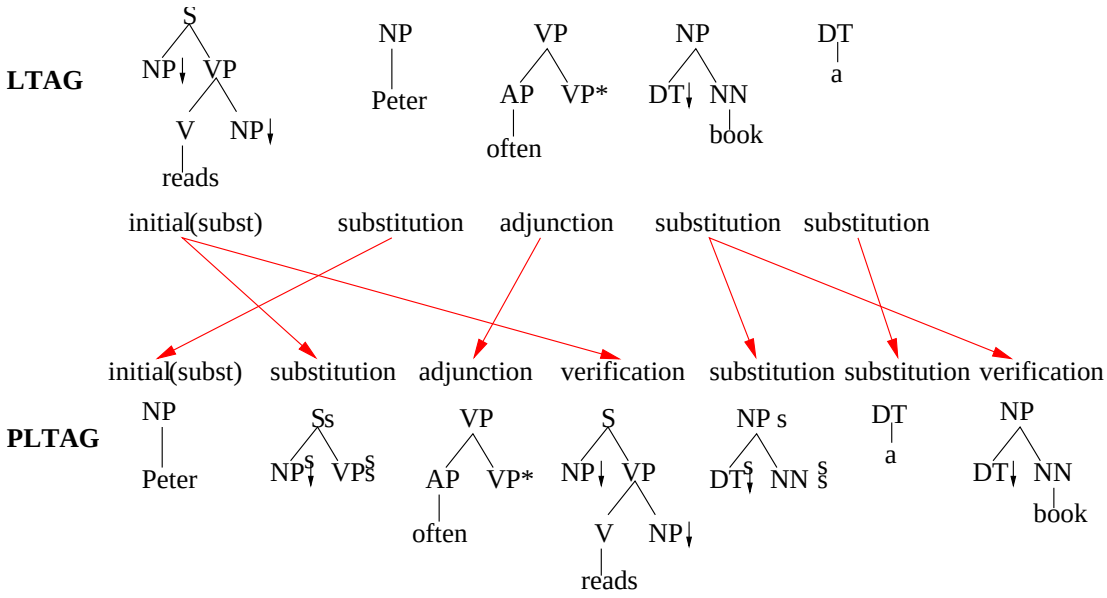


Figure 7.8: A LTAG vs. PLTAG derivation for the sentence *Peter often reads a book.* Note the relations between trees and operations in the LTAG vs. the PLTAG derivations, as indicated by the arrows: each LTAG derivation step that was in correct incremental order can be expressed by the same tree and same operation in the PLTAG derivation.

7.2.2 Equivalence of PLTAG and LTAG

Equivalence between two grammar formalisms means that both grammar formalisms assign the exact same structures to the same strings, and that they cover the same set of strings.

1. PLTAG should not over-generate with respect to LTAG.

2. PLTAG should not miss out on any analyses that LTAG can generate.

3. PLTAG should assign the same analysis to a sentence as normal LTAG would.

The first point, no over-generation, is easiest to show: since every predicted node has to be matched by a canonical node, only analyses that are exclusively made up of canonical nodes (and no remaining prediction nodes) are accepted. So the prediction trees do not introduce any additional structure, and hence cannot accept any sequences that are not accepted by LTAG.

The second point depends on the prediction lexicon. If we define the prediction lexicon such that for each tree in the canonical lexicon, there exists an exact copy with all nodes marked as predicted in the prediction lexicon, it is trivial to show that PLTAG can generate all analyses that LTAG can generate: We just order the LTAG trees by their lexical anchors and allow the prediction trees to be used when needed for connectivity. In the end they will all be verified by the identically-looking canonical tree. We are thus guaranteed to be able to obtain the same derived tree for the string of words.

The argumentation for the third point, whether analyses assigned to a sentence by PLTAG vs. normal LTAG are the same, is similar to the previous argumentations. It requires that *all and no more* analyses be found, and hence the same questions about prediction tree design are relevant as for the second point. [2]

Theorem: *For each LTAG grammar, there exists a PLTAG grammar such that the derived trees from the LTAG grammar are identical to the trees derived with the PLTAG grammar (given an adequate PLTAG prediction lexicon).*

[2] Note that in statistical processing (in particular if it also involves beam search) we cannot guarantee that the identical set of analyses remain in the beam, or that the analyses will be in the same order. Via the probabilities of the prediction trees, some analyses may be ruled out early on for being too improbable although they later turn out to be perfectly fine, or probabilities are different because of the distinct operations of integrating a prediction tree and then verifying it as opposed to directly integrating the canonical tree.

7.2.3 Predictions in PLTAG

In PLTAG, prediction occurs in two cases: when required by connectivity, and when required by subcategorization. At the end of this section, the exact shape of prediction trees used in this work is discussed.

7.2.3.1 Prediction through Connectivity

As we have seen in the examples 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 not the head of the parent node. This happens, for instance, at the integration of the second determiner in an ORC like *The senator that the reporter attacked, admitted the error*, as illustrated in Figure 7.9. 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*. 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 7.9).

Because natural language contains recursive structures, there are in theory infinitely many ways to connect two trees. Although embedding depth can be infinite in theory, we here assume that it is finite and indeed very small due to limitations of human memory. In our example in Figure 7.9, two prediction trees are needed to achieve full connectedness. As mentioned earlier in this chapter, predicted nodes are marked with unique indices, indicating which nodes should be verified by the same tree. The nodes that will eventually be verified by the *reporter*-NP tree have index $S2$, and nodes that will be verified by the tree anchored in *attacked* have index $S1$. Prediction trees can be pre-combined, like the one in our example, for efficiency reasons during parsing – this issue will be discussed in Section 8.4.1. Alternatively, two prediction trees, one containing the nodes with index $S1$ and the other one with nodes indexed $S2$ could be integrated into the prefix tree with two substitution operations.

7.2.3.2 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 yield a valid sentence. Each open substitution node that is to the right of the rightmost lexical anchor constitutes

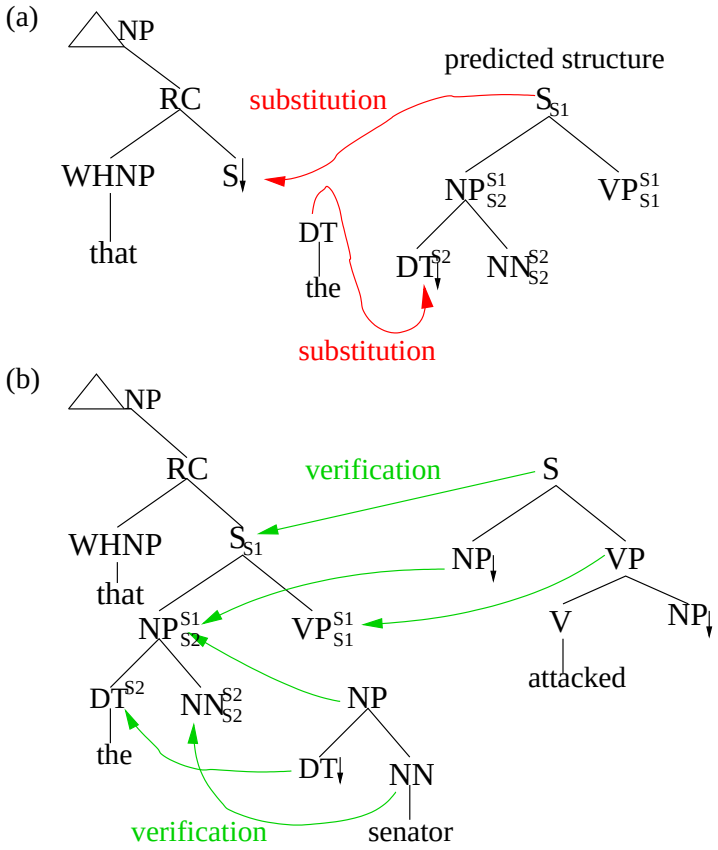


Figure 7.9: Prediction and Verification

a prediction during the parsing process. Modifiers are generally not predicted in our framework, unless they are needed for connectivity (see Section 7.3.3 for a more detailed discussion of this issue).

We exploit TAG's extended domain of locality in order to construct lexicon entries with more than one lexical anchor. We can use this to explain predictive facilitation for *either . . . or* and related constructions (Staub and Clifton 2006; for a more detailed discussion of generating such lexicon entries, see Section 7.3.2).

For the *either . . . or* case, we assign a lexicon entry to *either* which predicts the occurrence the conjunction *or*, as well as predicting a coordinate structure that combines two entities of the same category, see Figure 7.10(a).

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 7.10(c), (d)), leading to an ambiguity which will have to be resolved later on. For the sequence *Peter read either a book or* on the other hand, the word

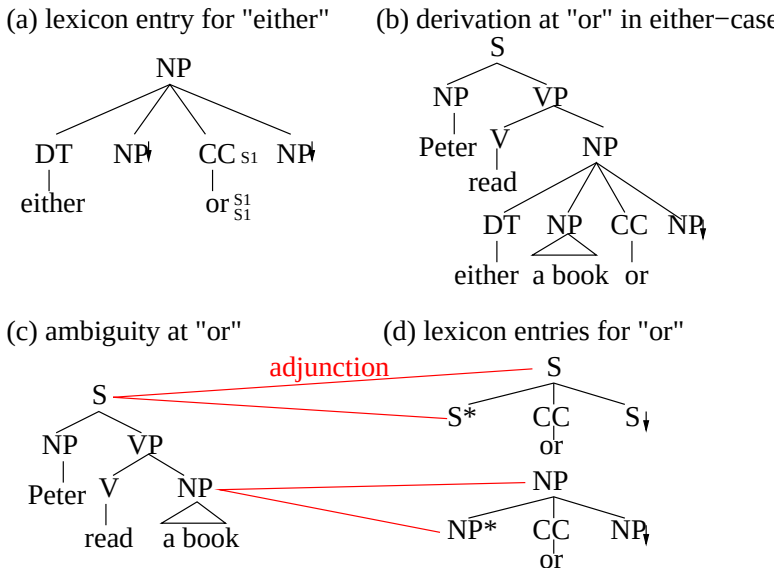


Figure 7.10: Extended domain of locality for expressions that trigger predictions.

or was predicted already at *either*, 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 7.10(b). A formal evaluation of this case is reported in Chapter 9, Section 9.1.2.

7.2.3.3 Controlling Prediction Granularity

The PLTAG formalism itself does not make any claim or pose any restriction on the shape of the trees in the prediction lexicon (except that they must contain more than one node, like all TAG trees). However, since each of these prediction trees will have to be verified by a tree from the lexicon later, any trees that would contain more nodes, or nodes that are in an arrangement that does not exist in the canonical lexicon, could never possibly be verified, and thus never lead to a valid PLTAG derivation. Therefore, it makes sense to only include prediction trees into the prediction lexicon that are the same or smaller than the canonical LTAG trees. Prediction trees that lack nodes to the right of their spine can still be verified by a canonical tree that includes those nodes (because the lack of those nodes has not possibly affected the derivation so far, due to the incrementality constraint).

Of course, this opens the question of how big exactly a prediction tree should be. The prediction tree sizes determine in fact the prediction granularity during parsing. If we decide to always use complete copies of canonical trees, just marked as predictions,

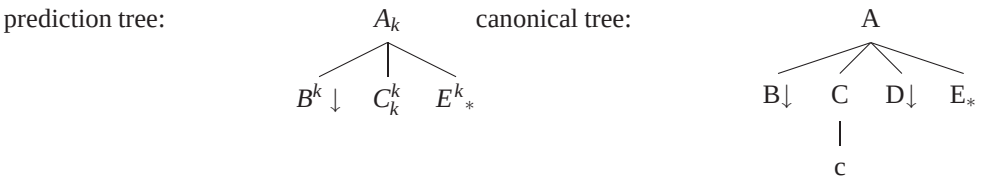


Figure 7.11: Example of a compatible prediction and verification tree, as defined for use in our system.

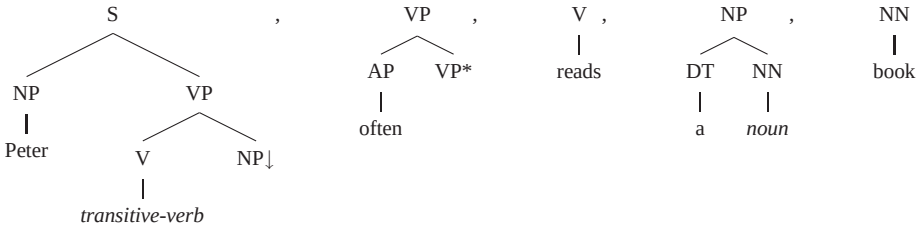
this would mean that we would always predict upcoming structures which are needed for connectivity down to the lexical item, and that we would for example predict full subcategorization frames for verbs before having seen the verb identity. This level of granularity would not only seem implausible psycholinguistically, but it would also mean that we predict much more detail than necessary for connectivity reasons, and would lead to a larger prediction lexicon. Instead, we will predict upcoming structures only as far as required by connectivity or subcategorization. (However, this is a preliminary assumption, as the optimal prediction grain size in remains an open research question.)

We therefore define that prediction trees have the same shape as trees from the canonical lexicon, with the difference that they do not contain substitution 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, for an example, see Figure 7.11. An exception to this rule are nodes that lie on the path between the root and the foot node. If a node to the left of the spine is missing in the prediction tree, it can not be matched against the verification tree, and hence not be verified by it as doing so would violate the incrementality assumption (the additional substitution node in the verification tree would not be filled at the time of processing the verification trees lexical anchor).

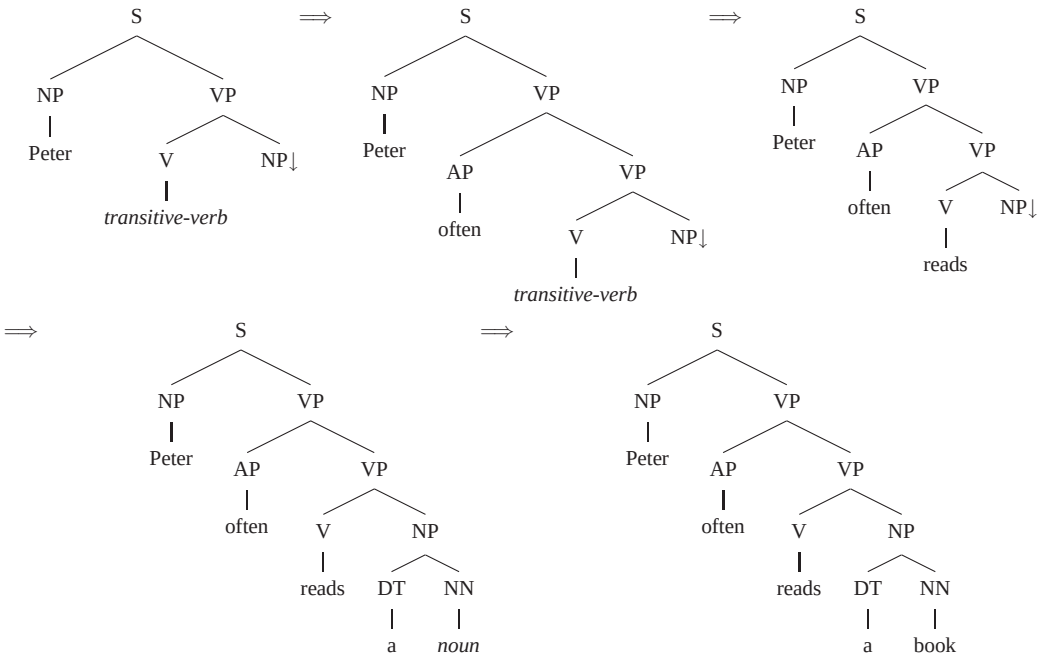
7.2.4 Comparison to DVTAG

The *Dynamic Version of TAG* (DVTAG) was developed by Alessandro Mazzei in his PhD thesis (Mazzei, 2005). Like PLTAG, it was motivated by constructing fully connected analyses. The problem was solved slightly differently in his version of the formalism: instead of predicting tree structures in a prediction step and having a prediction lexicon, the parts of the tree structures that would need to be predicted in PLTAG are pre-attached to the canonical lexicon entries, see Figure 7.12(a). The sentence *Peter often reads a book* would then be derived as shown in Figure 7.12(b).

The most important theoretical difference lies in the fact that what is predicted in PLTAG is not marked as predicted and hence is not subject to a verification mechanism in DVTAG. Furthermore, prediction granularity in DVTAG is different from PLTAG: in DVTAG, a larger number of more detailed alternative structures is generated. In his thesis, Mazzei states that 6 million tree templates were extracted by converting the 1,226 XTAG templates to DVXTAG templates. This means that the size of the grammar virtually explodes in DVTAG. This huge size of the grammar makes it very challenging, if not impossible, to implement a broad-coverage DVTAG parser.



(a) Some example lexicon entries in DVTAG.



(b) A DVTAG derivation.

Figure 7.12: A lexicon and derivation example in DVTAG.

7.3 Lexicon Design and Open Questions

This section discusses some general questions concerning the design of the lexicon, focusing on linguistic and psycholinguistic aspects (rather than implementational issues). As explained in the previous section, predictions depend on subcategorization frames of the lexicon entries. Furthermore, the shape of lexical trees also influences the degree to which additional structures for achieving connectivity are needed – if lexicon entries have the form as in DVTAG, connectivity is achieved without prediction trees.

7.3.1 Predicting a Sentence at the Beginning

An example for an open design issue that also affects the definition of a PLTAG derivation is whether a sentence node should always be postulated at the beginning. In psycholinguistic terms, are people always predicting that they are going to process a sentence? And do they do it in all situations? Both when reading a book and when in just casual discourse (where in fact many utterances are NPs or other fragments of sentences)? This would have consequences for the predictions required by connectivity, and hence the processing difficulty predicted by the linking theory. This issue also depends on the question whether e.g. the utterance of a single NP should be considered as just being an NP, or in fact an elliptic sentence. For the PLTAG derivation shown in Figure 7.8, always postulating a sentence would require changing the order of the first and second trees. More generally, it means that a verb-prediction tree (and possibly other structures like an NP structure if the first word is a determiner) would have to be predicted in all languages which are not verb-first. At the verb itself, verification cost would occur, meaning that a verb should be the more difficult, the longer the phrase before the verb. On the other hand, this might be out-weighed by the forward-looking component which expects a verb more and more strongly as the sentence unfolds.

In future work, it would be interesting to tease apart these two aspects and test them empirically.

7.3.2 Size of Lexicon Entries

At the *either..or* example in Figure 7.10, we have seen how lexical entries with two anchors (one of them being predicted) can influence the predictions. An important question is how to automatically and consistently decide which lexicon entries should have multiple anchors and which ones should not. Hand-selecting them without an

objective criterion will obviously lead to inconsistencies which may weaken claims about the predictions of the theory for naturally occurring broad-coverage text.

The problem of automatically learning the size of trees can be formulated in terms of data-oriented parsing (Bod et al., 2003). The criterion for deciding on tree size in a DOP framework are made based on the co-occurrence probabilities of words or inner nodes in a syntax tree. If a pair of words (like *either* and *or*) occurs together much more often than would be predicted by the "unigram" frequency of the words *either* and *or*, they will be encoded in the same tree. For the incremental framework it is thereby also particularly relevant which word is more predictive of the other. *Or* certainly is not as strong a cue for the occurrence of *either* as *either* is for *or*. So if the first word of such a pair would be highly predictive of the second one, we could define a threshold and include all those constructions into the lexicon as a tree for the first word which includes the predictive lexicalized entry of the second. At a more fine-grained level, this could also be done for internal nodes. A similar approach has recently been described by Cohn et al. (2009), who used a non-parametric Bayesian model for inducing Tree Substitution Grammars.

7.3.3 Arguments and Modifiers

Another question related to lexicon entry sizes is the distinction between arguments and modifiers. Currently there is mostly evidence of arguments being predicted: experiments targeted at detecting syntactic prediction usually try to show prediction of obligatory phrases or words, (e.g., Kamide et al., 2003; van Berkum et al., 1999b). The distinction between arguments and modifiers in practice is however often difficult to make for humans and laborious to annotate, indicating that the distinction is gradual rather than a categorial one. Even though we assume in our linking theory that arguments are predicted, while modifiers are not, some initial evidence (Arai et al., 2008) indicates that modifiers can be predicted in a context where they are required by discourse.

Distinguishing modifiers from arguments impacts statistical NLP in formalisms which make conditional dependence assumptions based on the argument / modifier distinction. For example, the probability of a specific prepositional phrase under a VP is dependent not only on the verb, but also on its other arguments, while the probability of an adverbial phrase that modifies the verb (like *tomorrow*) may be independent of the verb's other dependents.

7.3.4 Free Word Order Languages

The plausibility of our "minimal" predictions which only predict arguments that come before the head of a phrase, but not the ones after can also be questioned. In English, the syntactically most obligatory argument, the subject, is positioned before the verb, while all other arguments usually occur after the verb. Minimal predictions thus provide a convenient way for generalising over intransitive vs. transitive vs. ditransitive verbs. In languages that have more head-final constructions, and / or where several arguments can be arranged in any order before the verb, we would end up making more precise predictions and thus change the prediction granularity level with respect to English. This does not seem very desirable.

A possible solution would be to assume a multi-set representation for these arguments in free word order languages, rather than postulating many alternative structures, as has been suggested for CCG (Hoffman, 1995; Baldridge, 2002). A multi-set representation would mainly affect lexicon size and difficulty predictions made by the linking theory due to the prediction and verification mechanism. In the verification mechanism, the function that tests for compatibility between a predicted tree and a verification tree would have to be modified such that the verification tree can match a compatible multi-set representation.

7.3.5 Traces

The notion of traces has its origin in Government Binding theory. Traces are phonetically empty elements that are connected in the syntactic structure. Traces are used in some syntactic theories to account for e.g. wh-movement and passives. However, the existence of traces is controversial. See (Sag and Fodor, 1994) for a detailed discussion of linguistic as well as psycholinguistic evidence for and against the existence of traces. This thesis assumes the existence of traces.

In the final version of this work, only traces encoding A'-movement (including parasitic gaps), passives, control verbs and null complementizers were used, while placeholders for ellipsed material, right node raising, expletives, and other pseudo-attachments marked in the Penn TreeBank were ignored. Treatment options for traces in relative clauses, passive constructions, rising and control constructions, extractions (including long-distance extractions) and parasitic gaps are discussed in Appendix B.

7.4 Conclusions

This chapter suggested new variant of tree adjoining grammar, called Psycholinguistically Motivated TAG (PLTAG). Its most important properties are that it allows for strictly incremental derivations, and supports a psycholinguistically more plausible prediction grain size than DVTAG. PLTAG models the processes of prediction and verification explicitly, and for doing so introduces prediction trees, which are similar to canonical TAG trees but can be unlexicalized trees, and a new operation called verification.

We have shown the equivalence between PLTAG and standard LTAG, and explained how the formalisms can be mapped onto one another. We then addressed some design questions that are relevant for using the formalism in a specific setting, such as with a particular grain size for prediction trees, or application for languages that have different properties, such as more flexible word order, from English.

The formalism was designed to meet the specifications of the sentence processing theory outlined in Chapter 6. An implementation of the formalism, including an automatically converted tree bank, automatically induced canonical lexicon and prediction lexicon and a probabilistic, broad-coverage parser will be presented in Chapter 8.

Chapter 8

An Incremental Predictive Parser for PLTAG

The incremental predictive parser proposed in this chapter implements the restrictions and requirements lined out in Chapters 6 and 7. This chapter describes the conversion of the Penn Treebank to a format that is compatible with PLTAG, the induction of a lexicon from the transformed treebank and the design of the parsing algorithm and its probability model. We found that in practice, the lexicon extracted from the Penn Treebank did not contain any TAG trees that were not TIG[1] trees. Similarly, other recent parsers like the Chiang (2000) parser is only a TIG parser, and recently, CCG bank (Hockenmaier and Steedman, 2007) and the C&C parser (Clark and Curran, 2007) have been shown to parse using exclusively context-free rules (Fowler and Penn, 2010). To parse English based on the context-free lexicon extracted from the Penn Tree Bank, it is only necessary to handle trees that also satisfy the condition of being TIG trees. Therefore, the PLTAG implementation described in this chapter is actually only an implementation of a PLTIG parser.

In the last part of this chapter, we evaluate parsing performance. The parser described here is to our knowledge the first fully incremental and predictive parser. Finally, the linking theory, which maps the parser actions to processing difficulty, is formalised and its implementation described.

[1] TIG stands for Tree Insertion Grammar, for more information see Section 6.3.3.

8.1 Treebank Conversion

In order to implement a parser for PLTAG, the first step is to create a resource of PLTAG syntax trees, which can be used for training and testing the parser (training data is needed because we here describe a supervised parser for PLTAG. Supervised approaches generally achieve higher accuracy than unsupervised ones). An existing big tree bank is the Penn Treebank, however its format is slightly different from PLTAG trees, mainly in that tree structures are flatter than TAG trees. The Penn Treebank structures have already been converted to TAG structures in previous work by (Xia et al., 2000), whose procedures we follow in this work. We decided to convert the Penn TreeBank ourselves instead of using an existing converted TAG treebank in order to to add the NP annotation from Vadas and Curran (2007), and extract the prediction tree lexicon more easily.

As a first step, the NP annotation from Vadas and Curran (2007) was added to the Penn Treebank annotation, thus disambiguating the flat NP structures. Next, dependents were marked as either arguments or modifiers of a head, following annotation from PropBank (Palmer et al., 2003). Finally, all nodes in the tree were marked as to whether they are the head child of their parent node, using a slightly modified version of Magerman's head percolation table (Magerman, 1994). This section will discuss how the flat structures from the Penn Tree Bank were disambiguated in order to be proper PLTAG derived trees, and how special cases like auxiliaries and copula constructions were handled.

After conversion, the resulting PLTAG treebank can also be used to automatically induce the canonical lexicon and the prediction lexicon, as reported in Section 8.2.

8.1.1 Disambiguating Flat Structures

The structure of the Penn Treebank is flatter than the typical structure of PLTAG derived trees. In PLTAG, a new internal node is introduced (via the root and foot node of an auxiliary tree) whenever an adjunction operation takes place. Therefore a PLTAG tree has typically many binary branches where the Penn Treebank uses flat structures. We assume right branching structures (for an example, see Figure 8.1), following previous efforts of converting the Penn Treebank into binary formats (Hockenmaier and Steedman, 2007). However, the heuristic of right-branching does not always lead to correct results, and has been shown to be particularly problematic for NPs.

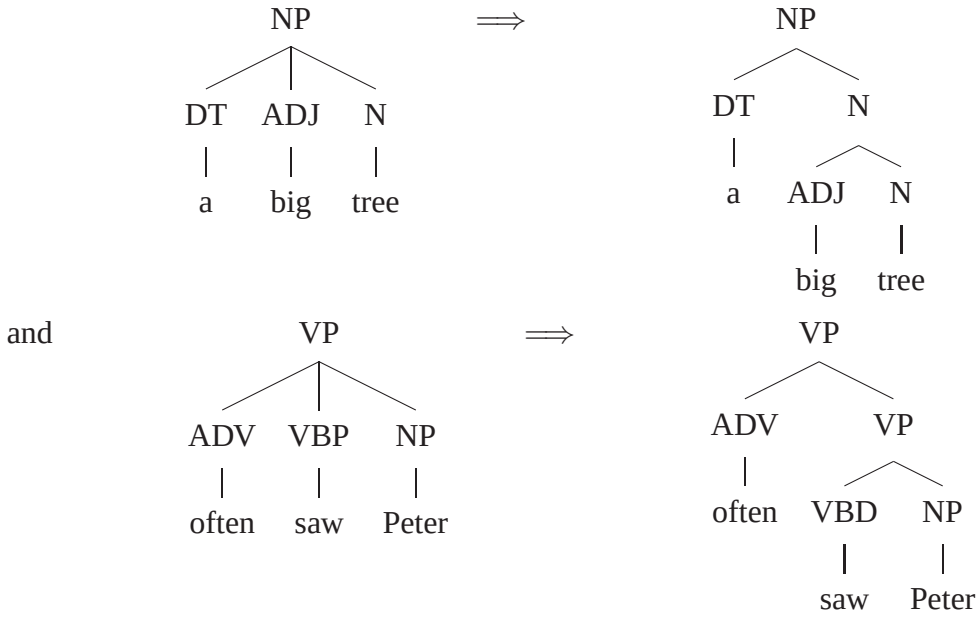


Figure 8.1: Binarisation of flat Penn Treebank structures into right branching binary structures for PLTAG.

Noun phrases (NPs) and quantifier phrases (QPs) are usually assigned a completely flat structure in the Penn Treebank. While the noun phrase annotation by Vadas and Curran (2007) has remedied this to a certain extent by introducing disambiguating nodes to mark left branching inside NP phrases, right branching remains implicit, and there is thus an asymmetry in the annotation for left vs. right branching. We therefore introduce additional nodes in NPs to denote right branching, see Figure 8.2.

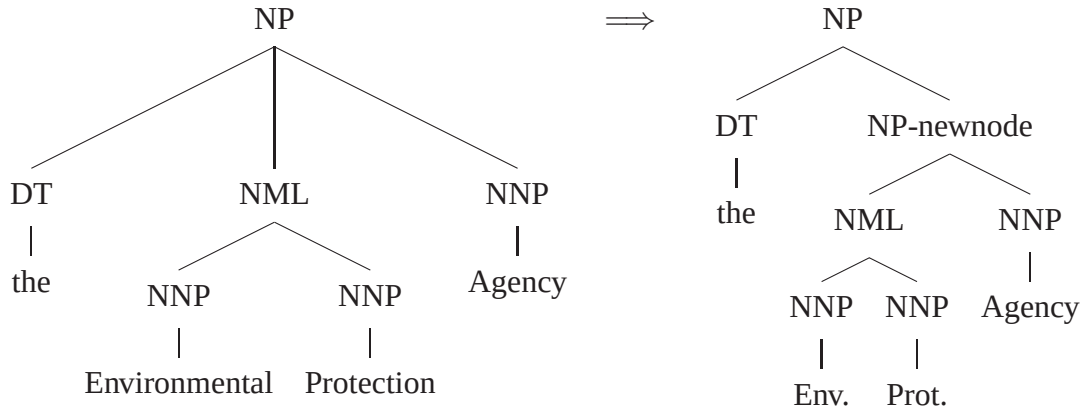


Figure 8.2: Introducing explicit right branching.

Furthermore, quantifier phrases were unfortunately not disambiguated in the NP

annotation, even though they suffer from the same problem of being annotated very flatly (see example tree structure in Figure 8.3). Heuristics are used to cope with these cases.

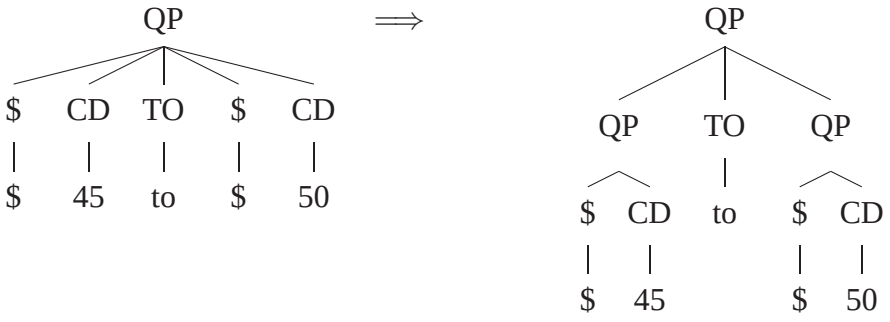


Figure 8.3: Structural disambiguation of Quantifier Phrases.

The most common case of inaccurate right branching structure remaining in the treebank even after NP annotation concerns coordinated structures. We try to recognise scopes within coordination automatically and introduce the missing nodes (see Figure 8.4). For sentence-initial modifiers such as *but*, *and*, a new POS tag was introduced (CCSIM), in order to distinguish this case from proper coordination within a sentence. If we were operating more on a discourse level, these sentence-initial conjunctions could be handled in the same way as normal conjunction.

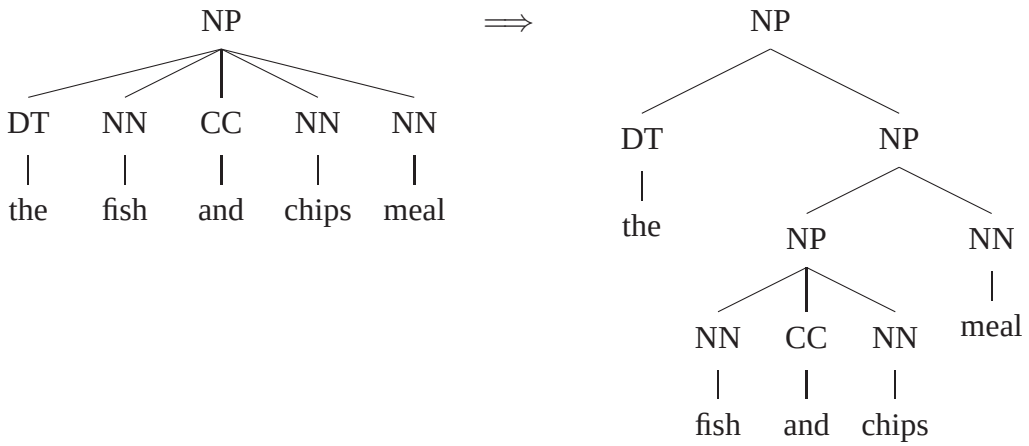


Figure 8.4: Structural disambiguation of coordinated phrases.

8.1.2 Auxiliary Treatment

In the Penn Treebank, auxiliaries and modals have the same part-of-speech tag as full verbs. Following the standard head percolation rules, they are therefore determined to

be the head of phrases since the algorithm is not able to distinguish them from regular verbs. A heuristic for detecting auxiliaries and modifiers was implemented in order to assign them a different POS tag 'AUX', which then enables the lexicon induction algorithm (see Section 8.2) to encode them as an auxiliary tree, see Figure 8.5.

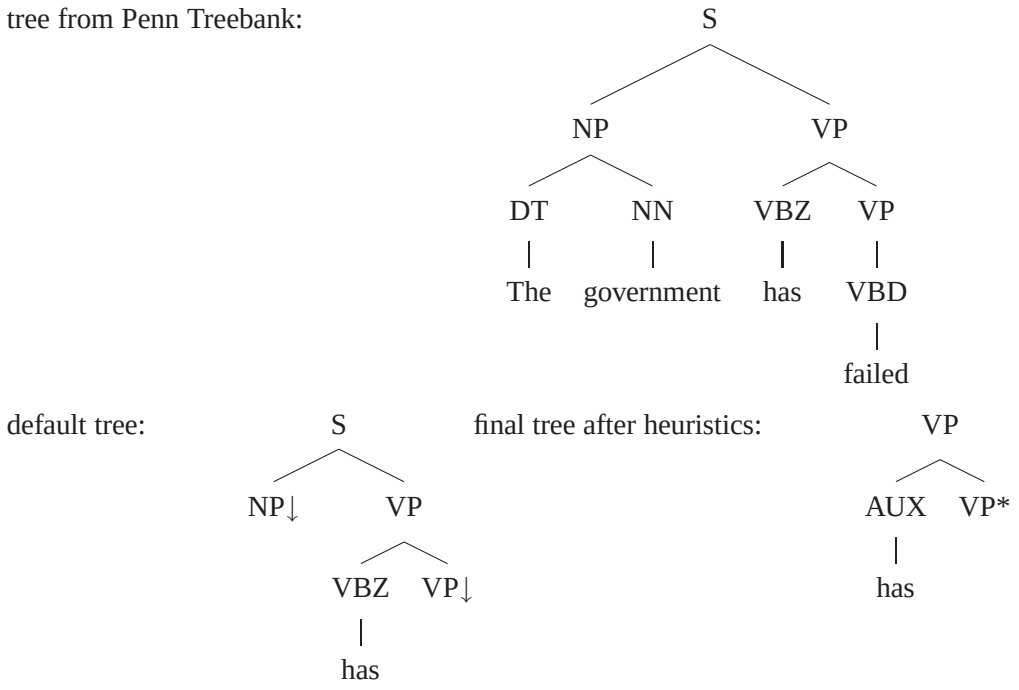
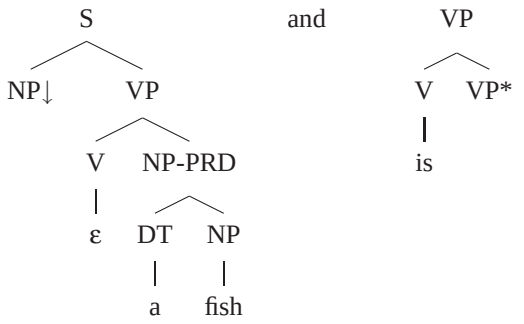


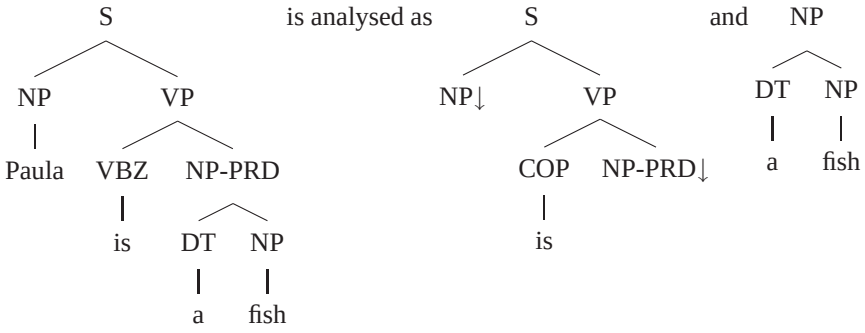
Figure 8.5: Auxiliaries and modals are assigned a special POS-tag 'AUX' to distinguish them from full verbs and correctly extract an auxiliary tree template for them.

8.1.3 Copula Treatment

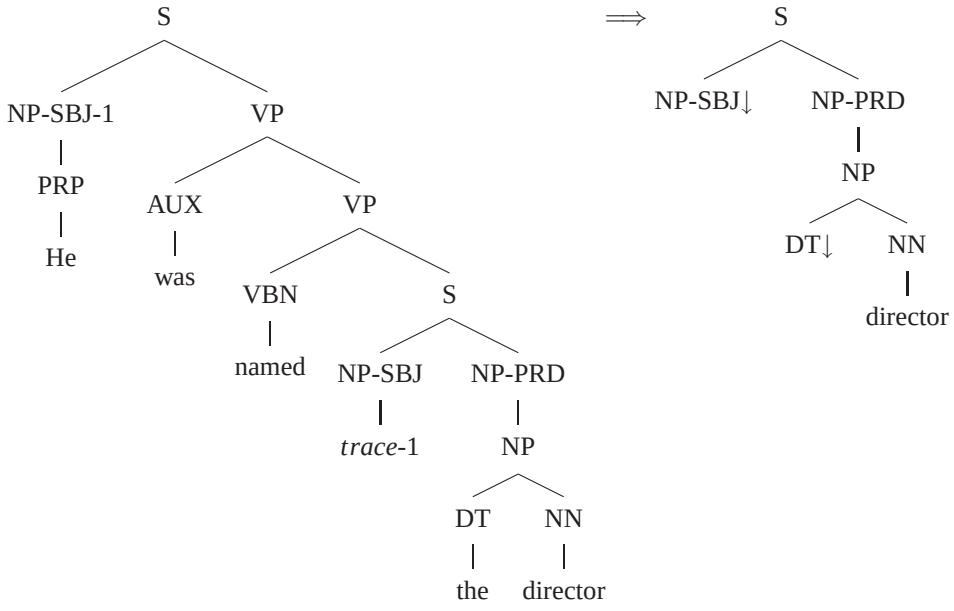
The standard XTAG analysis of copula constructions introduces a new initial tree for each predicate, and an auxiliary tree for the copula, see Figure 8.6(a). We did however not adopt the standard XTAG analysis for PLTAG, because it would lead to a larger number of lexicon entries: every predicate would be annotated with the full S-NP-VP-structure. Instead, the predicate noun is assigned the typical NP tree template, see the tree for *fish* in Figure 8.6(b). This can be achieved by introducing a special POS-tag for copula verbs. However, there are some cases, where the annotation of the treebank forces us to assign the XTAG-entry structure to the NP: In Figure 8.6(c), the word *director* must be assigned a sentence structure because no trace or null element for the copula is annotated.



(a) XTAG analysis of copula constructions.



(b) PLTAG analysis of copula constructions.



(c) Exception case for PLTAG analysis of copula constructions.

Figure 8.6: Treatment of copula constructions in PLTAG.

8.1.4 Remaining Problems

The treebank conversion algorithm assigns complete structures to ca. 97% of all trees in the Penn Treebank. Some of the remaining three percent of trees were only partially converted and may miss some leaves (this mainly affects sentences that include FRAG (for fragment) nodes. Among the 97% of complete conversions, there is a small number of sentences where punctuation marks or some modifiers are in the wrong order in the tree, due to the fact that modifiers in TAG cannot adjoin to a node between two arguments of a node.

8.1.4.1 Modification occurring between two arguments, or between a head and its argument

The biggest cause of not being able to segment the tree correctly are modifiers which occur in-between two arguments, or between a head and its argument (see Figure 8.7). In standard TAG, such cases are handled by introducing additional VP nodes that the modifiers could use as an adjunction site. In our implementation, we inserted a VP node above each VB, VBD, VBG, VBN and VBZ node that is directly dominated by a VP. This step creates additional attachment sites, and can thus have a potentially negative effect on precision of a parser using these structures. However, the introduction of these additional nodes improves coverage substantially, because it then allows to derive sentences with modifiers that occur in-between arguments. We found that on the training data, the introduction of these additional VP nodes reduced trees that could only be transformed to a wrong attachment order by more than 20%.

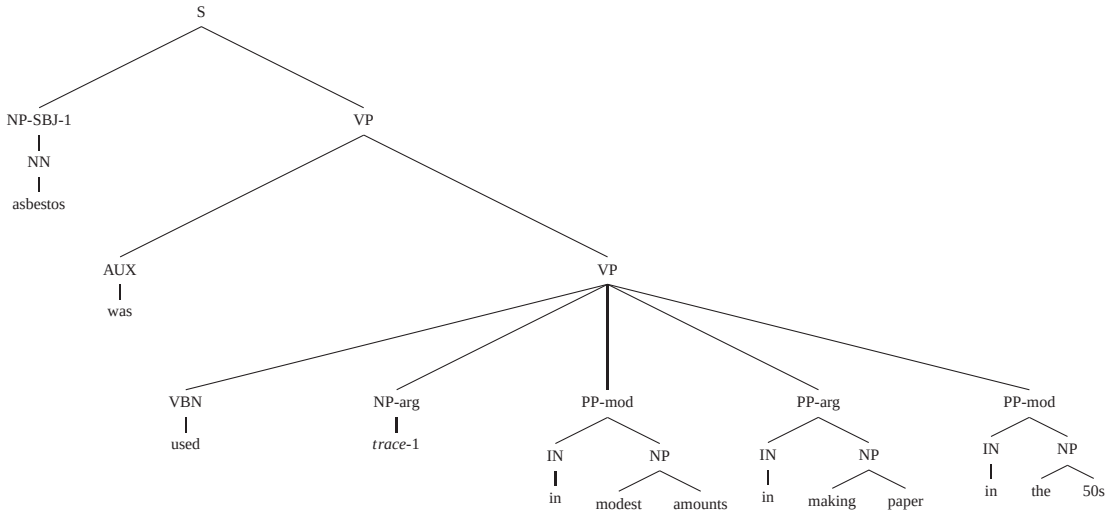


Figure 8.7: Problematic case of a modifier occurring between a head and its arguments.

8.1.4.2 Punctuation Disambiguation

Another tricky case is punctuation. In the literature about parsing, punctuation is usually raised in the tree as far as possible, or simply removed. A reason why punctuation treatment is difficult is that it is often used inconsistently (this is in particular also true for the Penn Tree Bank data). However, punctuation does contain useful information, and it has been shown that it is beneficial to use it at least in some form (e.g. as a feature) to inform the parser. As this thesis focuses on modelling human processing and evaluation on text which contains all punctuation marks, we decided to try to keep punctuation marks if possible. However, the question was then whether they should be modifiers or arguments. Treating all punctuation marks as arguments helps to prevent the problem of having a modifier (the comma) between two arguments. On the other hand, treating punctuation as arguments leads to a much larger lexicon, and poor generalisation performance. For example, sentence-final punctuation is problematic as an argument to the sentence for sentence-level coordination, causing the lexicon entries for verbs not to generalise across sentence-level coordination vs. uncoordinated phrases. Removing any sentence-final punctuation, as well as brackets, quotation marks and dashes is thus an effective way to significantly reduce lexicon size. It is a well-established way of dealing with punctuation, see also (Collins, 1999; Bikel, 2004).

In the final version of the treebank converter, heuristics are used to identify apposition-like insertions and introduce a new node with category 'APP' with the first comma as its head, and subcategorising the apposition and second comma (see Figures 8.9 and 8.8). All other punctuation was removed if at the beginning or end of a sentence, or treated as a modifier.

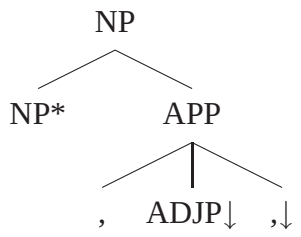


Figure 8.8: Treatment of commas in appositions.

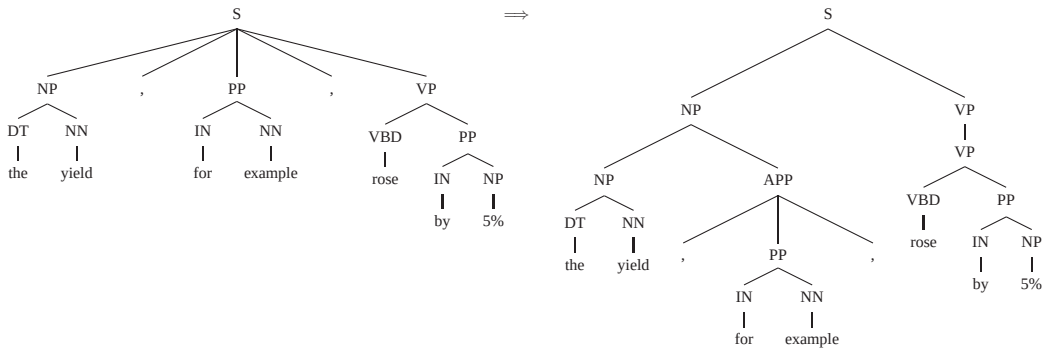


Figure 8.9: Example of three modifiers (two commas and a PP) occurring between the argument and a head. Correct order can be achieved by introduction of an additional node 'APP', and adjoining the apposition with either the NP or the VP.

8.2 Lexicon Induction

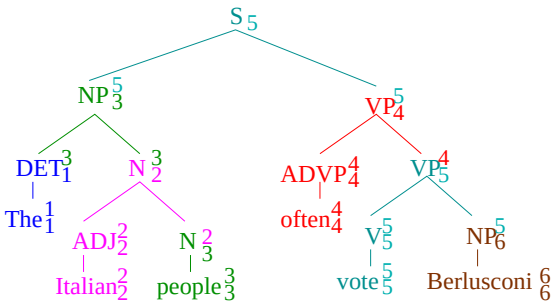
After converting the Penn Treebank into PLTAG format, and adapting some of the annotation as described in Section 8.1, the resulting PLTAG treebank can be used to induce a PLTAG lexicon. As described above, we annotated the transformed structures with head information from an adapted version of Magerman's (1994) head percolation table. The head information is necessary to segment the tree for the sentence into the two types of lexicon described in Chapter 7: the canonical lexicon entries, and a prediction lexicon (similar to the canonical lexicon, but mostly not lexicalized).

8.2.1 Creating the Canonical Lexicon

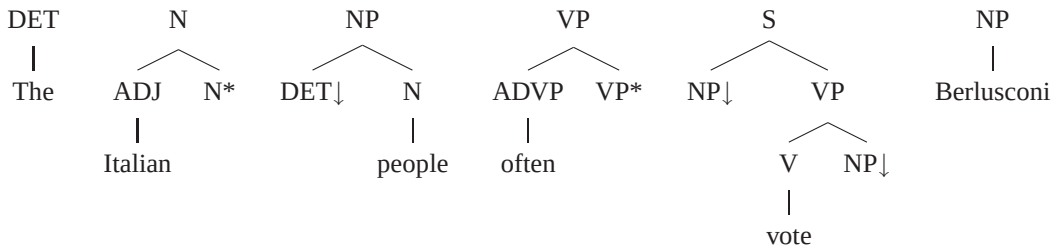
The creation of the canonical lexicon is based on the procedure described in (Xia et al., 2000). Each node in a tree from the PLTAG tree bank must be annotated with a flag indicating whether it is its parent's head child or not. Furthermore, each lexical anchor must be annotated with a flag stating whether it is a modifier or an argument.

The algorithm then determines the spine of an elementary tree by starting at each lexical leaf of the tree and checking its ancestor nodes for whether they are the head child or not. Whenever a node is encountered that is not the head child of its parent, the algorithm checks whether the lexical anchor is a modifier or an argument. If it is an argument, the node is cut into two halves. The upper half is marked as a substitution node, and the lower half constitutes the root of the elementary tree. For example, consider the sentence *The Italian people often vote Berlusconi* shown in Figure 8.10(a). The algorithm would for example check the parent of *The* for whether it is the head

child of its parent node *NP* and find that it is not. It would then check whether *The* is an argument or a modifier and find that it is an argument, and therefore cut the *DET* node into two halves, thus creating the substitution site in the elementary tree for *people* and the root node for the elementary tree of *The*, see Figure 8.10(b).



(a) PLTAG tree showing which part of the tree will result in which lexicon entry.



(b) Lexicon entries extracted from the above tree.

Figure 8.10: Generating lexicon entries from the PLTAG tree bank.

If the lexical anchor is a modifier, the parent node is cut in half (the lower half is going to become the elementary tree's root node), and the future foot node has to be identified by checking the parent node's head child (and its head children) for a node of same category as the parent. The foot node is then also cut in half. The upper half constitutes the new elementary tree's foot node, and the lower half is joined with the parent node's upper half to form a complete node. This happens at the words *Italian* and *often* in Figure 8.10(a). At the word *Italian*, the *ADJ* node is not the head child of its parent *N*. As *Italian* is a modifier, we cut the parent node *N* into its upper and lower half, and find that its head child also has category *N*, and also cut it in half. The colours and indices in Figure 8.10(a) mark which part of which node will eventually belong to which elementary tree. Figure 8.10(b) shows the elementary trees extracted from the example sentence.

8.2.2 Creating the Prediction Lexicon

As defined in Chapter 7, prediction structures can be unlexicalized, and can basically have any shape. As discussed in Section 7.2.3.3, only those prediction trees that match the structure of some canonical tree can yield valid PLTAG derivations. Therefore, only the subset of possible trees which are in the lexicon, or pruned versions of them can be of any use to the parser. We can further restrict this set by defining the desired level of generalisation (also discussed in Section 7.2.3.3). The prediction trees used in this work include the tree structure to the left but not to the right of the spine, cutting off any unary nodes at the bottom of the spine.

One way of generating such trees would be to transform the entries from the canonical lexicon. But a large set of prediction trees risks to make the parsing algorithm very slow because it creates a very big search space. We therefore restrict the set of prediction trees to structures that turn out to be necessary for incrementally parsing the PLTAG tree bank. As mentioned earlier on, prediction trees are needed for connectivity whenever two dependents precede a head, or when a grandparent and a child have been encountered, but the head of the parent node has not been seen. In our example sentence this happens at the words *Italian* and *often*. A systematic way to find these cases is by calculating the *connection path* at each word of the tree, and then subtracting the nodes from all elementary trees whose lexical anchor has not been seen.

Connection paths were defined and used for calculating connectivity for DVTAG by Lombardo and Sturt (2002). A connection path for words $w_1 \ldots w_n$ is the minimal amount of structure that is needed to connect all words $w_1 \ldots w_n$ into the same syntactic tree. The amount of structure needed at each word for the sentence *The Italian people often vote Berlusconi* is indicated in Figure 8.11 by the structure enclosed in the circles.

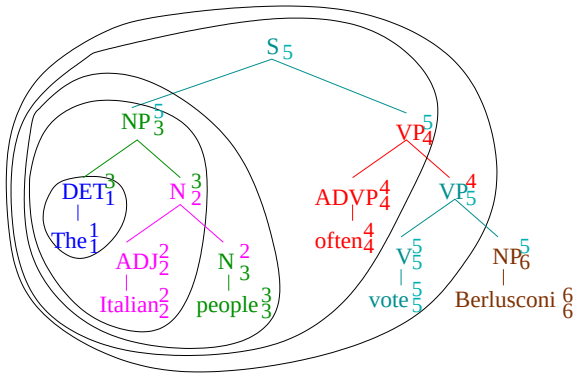


Figure 8.11: The connection path at each word.

We then use the connection paths and our knowledge of how the sentence tree can be decomposed into 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 which are not part of any of the elementary trees with feet $w_1 \dots w_n$. In Figure 8.11, this occurs twice: firstly when *Italian* has been read, and the determiner and adjective can only be combined by predicting that they must be part of the same noun phrase, and secondly at *often*, when the VP and S nodes have to be predicted.

As can be seen in Figure 8.12, there is some structure needed for the connection path which is part of an elementary tree whose lexical anchor has not yet been processed. The nodes required by the connection path but which are not part of elementary trees with already seen anchors constitute the prediction tree, see Figure 8.12. These prediction trees differ from the trees in the canonical lexicon in that all their nodes are marked as prediction nodes, and in that they are not necessarily lexicalized.

It can happen that nodes from two or more different elementary trees are needed by the connection path. In this case, we generate a pre-combined prediction tree (see Section 8.4.1). A pre-combined tree has unique indices for nodes that originate from different elementary trees, and is equivalent to generating several single prediction

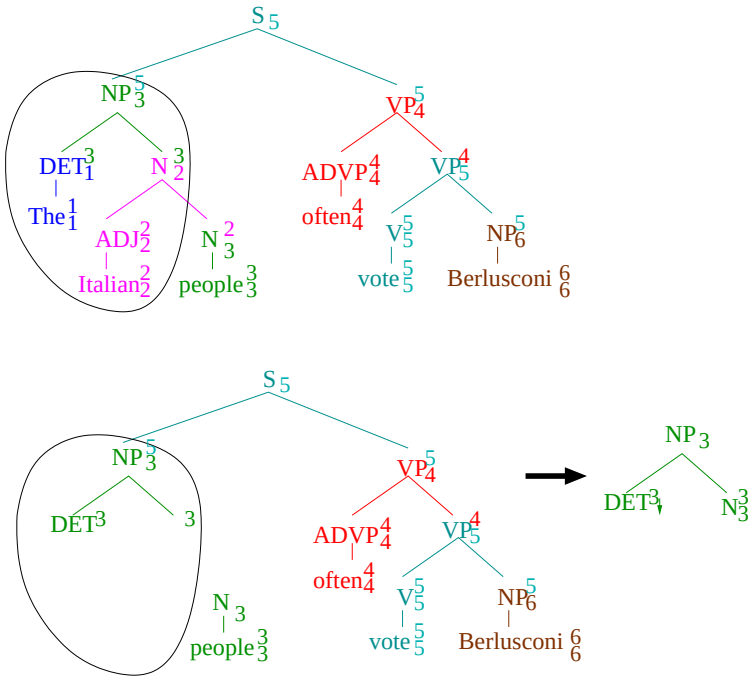


Figure 8.12: "Subtracting" the seen lexicon entries for *the* and *Italian* from the connection path structure at the word *Italian* leaves us with the connection bit, from which we generate the prediction tree, as shown on the right hand side.

trees and integrating them later online.

The prediction trees generated by the connection path method are simplified by removing unary nodes that were originally introduced through adjunction from the right side. This makes the trees smaller, reduces lexicon size and prevents us from predicting the exact points of modification in a tree.

8.2.3 Lexicon Induction Statistics

Based on the argument/modifier decisions based on PropBank and special treatment of punctuation our extraction algorithm generated 6000 prediction tree templates and 17000 canonical tree templates (i.e. unlexicalized trees). There were 146k unique canonical lexicalized trees (counting simplified categories, e.g. using NP instead of NP-SBJ). We obtained these numbers after trying to make the lexicon more compact, by treating all relations annotated as "Support" in PropBank as modifiers, and also treating as arguments words annotated as "-(VOC), (DIR), (LOC), (MNR), (PRP), (TMP), (CLR)" in PropBank. The number of tree templates extracted in this first version of the converter is considerably higher than the number of tree templates extracted in related work (Xia et al. (2000); Chen (2001) extracted about 6000 templates). This gap is due to differences in the treatment of punctuation and traces, as well as differences in the set of non-terminal categories used. The large lexicon led to very low coverage of the grammar ($< 60\%$). In order to increase coverage, the following changes were made:

- using fewer categories (NN, NNS, NNP, NNPS $\rightarrow$ NN; VB, VBD, VBG, VBN, VBP, VBZ $\rightarrow$ VB; JJ, JJ, JJR, JJS $\rightarrow$ JJ; RB, RBR, RBS $\rightarrow$ RB; NML, NAC, NP, NX $\rightarrow$ NP; WDT, WP, WP$ $\rightarrow$ WP; PRP, PRP$ $\rightarrow$ PRP)
- treating all punctuation marks except when in apposition-like construction as modifiers
- removing sentence-initial and sentence-final punctuation marks
- removing all traces and null-elements except '0', '*', '*T*'
- removing the top empty category of the treebank sentences

After these measures, the size of the extracted lexicon was more than halved to 7100 tree templates (thereof 2000 unique ones) and 2800 prediction trees, achieving a coverage of more than 90% on Section 23 of the converted Penn Tree Bank (unseen during

training). Lexicon size can be further reduced by using sister adjunction instead of the adjunction operation, as in (Chiang, 2000), whose extracted lexicon contained only 2100 tree templates, and by introducing commas as heads of auxiliary trees in coordination, as done in (Chen, 2001). For a good analysis of different extraction strategies and their effects on lexicon size see Chen (2001).

The average ambiguity per lexical item is 2.45 trees per word for the later version of the lexicon. The distribution is not even but follows the Zipf distribution as can be expected for language data. There are a few words with lots of different trees (in particular common function words like “and” (578 trees), “or” (219 trees), “as”, “in”, “but”, “is”, “$”, “of”, “for”, “to” (between 100 and 200 different trees each, see Figure 8.13).

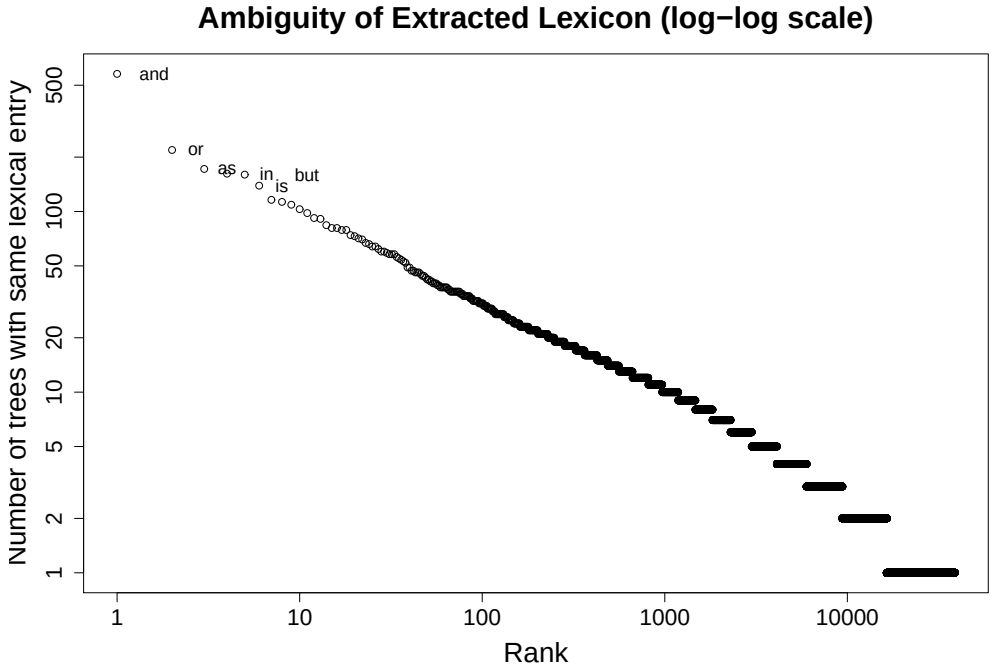


Figure 8.13: The distribution of ambiguity in the PLTAG lexicon.

Among the derivations where prediction trees were needed in order to achieve connectivity, 89.3% of cases used one prediction tree (at a time, without intervening canonical trees), in 10% of cases, 2 prediction trees had to be combined before connectivity was achieved, and in less than 1% of cases were three prediction trees needed. There were no instances in the Penn Treebank where more than 5 trees were needed.

8.3 The Incremental Parsing Algorithm

Given the lexicon entries and the PLTAG treebank for training and evaluation, we next describe the incremental parsing algorithm. The parsing strategy of the incremental PLTAG parser is related to an Earley parser. Alternative strategies, for example top-down parsing (Lang, 1991) would not lead to incremental PLTAG derivations as the parser would have to start with the sentence node, whose lexical anchor is the head of the sentence; and the head of the sentence is most often is not the first word of a sentence. Bottom-up TAG parsing (Vijay-Shankar and Joshi, 1986) on the other hand, would not lead to fully connected structures.

The implementation of the parser uses a chart to store and retrieve the parallel partial analyses at each word (see Section 8.4.3 for more details on the chart). The chart allows to aggregate similar analyses during the search phase and only calculate complete derived trees when search is completed for the n-best trees. To limit the size of the chart (both in terms of the number of different chart entries and the number of analyses in one chart entry), a beam is implemented to only keep the best analyses and prune analyses with low probability. As a further measure to increase efficiency, a supertagging step (see Section 8.4.5) for choosing prediction trees is interleaved with the parser steps. The supertagger selects the most promising prediction trees given the context, such that the parser only has to try to integrate the *n* best prediction trees instead of all trees from the prediction lexicon at each step, which dramatically cuts down on the search space.

The high level parsing algorithm (shown in Algorithm 1) works as follows: When processing a new input word w_i, the algorithm retrieves all possible elementary trees ε_{w_i} for w_i (line 2) and tries to combine them with all analyses covering the prefix of the sentence $\beta_{w_1..w_{i-1}}$ (line 6). It then retrieves a subset of the prediction trees (line 16–18; the subset is determined by the a super-tagger, see Section 8.4.5), and tries to combine the prediction trees with all analyses covering words $w_1..w_i$ (line 20). The prefix trees are stored in a chart (see lines 2, 13), which will be discussed in Section 8.4.3. To cut down on the search space and only follow the most promising analyses, prefix trees that have too low probability are pruned (lines 9 and 15).

The operations for combining the trees consist of adapted versions of the standard LTAG operations plus a verification operation. Note that we found that all auxiliary trees that we extracted from the Penn Tree Bank satisfy the definition of TIG trees, i.e. their foot nodes are the innermost or outermost leaf of the auxiliary tree. We will

Input: canonical and prediction lexicon,
input sentence $w_1 \dots w_n$,
frequency count from training
Output: prefix tree for words $w_1 \dots w_n$

1 **foreach** *word* w_i *in* $w_1 \dots w_n$ **do**
2 retrieve prefix trees $\{\beta_{w_1 \dots w_{i-1}}\}$ from chart
3 get elementary trees $\{\varepsilon_{w_i}\}$ for w_i from lexicon
4 **foreach** $\beta_{w_1 \dots w_{i-1}}$ **do**
5 **foreach** ε_{w_i} **do**
6 try to combine $\beta_{w_1 \dots w_{i-1}}$ with $\varepsilon_{w_{i+1}}$
7 **end**
8 **end**
9 prune resulting trees $\beta_{w_1 \dots w_i}$
10 **foreach** $\beta_{w_1 \dots w_i}$ **do**
11 expand future fringes if necessary
12 combine with trees with same fringe
13 insert into chart
14 **end**
15 prune again
16 get prediction trees $\{\pi_k\}$ from lexicon
17 **foreach** $\beta_{w_1 \dots w_i}$ **do**
18 select *n* best π out of $\{\pi_k\}$ (super-tagging)
19 **foreach** π *out of the selected ones* **do**
20 try to combine $\beta_{w_1 \dots w_i}$ with π
21 **end**
22 **end**
23 repeat lines 9-15 for $\{\beta_{w_1 \dots w_i}\}$ and $\{\pi_k\}$
24 **end**

Algorithm 1: The PLTAG parsing algorithm.

keep referring to the formalism as PLTAG in this thesis, even though the operations defined in our parser only handle (and for English only need to handle) PLTIG trees.

As we have defined in Section 7.2.1, a PLTAG derivation starts with the first word of the sentence. Partially derived trees are always prefix trees, i.e., they span a prefix of the sentence. The upcoming elementary tree may substitute or adjoin into the prefix tree (as in standard TAG), but also vice versa. We distinguish these cases by having `Up` and `Down` versions of the parser operations. The elementary tree to be integrated with the prefix tree can either be a prediction tree `P` or a canonical tree `F`. For example,

manner	element. tree into prefix tree		prefix tree into element. tree		if matching pred
operation	Substitution	Adjunction	Substitution	Adjunction	Verification
canonical	SubstDownF	AdjDownF	SubstUpF	AdjUpF	verif
predict	SubstDownP	AdjDownP	SubstUpP	AdjUpP	NA

Table 8.1: Parser operations table for an incremental PLTAG parser.

AdjDownP is the parser operation that adjoins the prediction tree into the prefix tree, while SubstUpF substitutes the prefix tree into a canonical elementary tree, see a list of all operations in Table 8.1.

The parsing algorithm does not allow two prediction operations to be executed in a row, to avoid an overly large search space. Cases where nodes from more than one elementary tree need to be predicted to achieve connectedness (as in the case of left recursion) are covered if they have been seen during training and are thus available as pre-combined prediction trees in the lexicon (see Section 8.4.1). The verification operation can only be applied if the prefix tree contains predicted nodes which match the structure of the canonical elementary tree.

8.3.1 The Concept of Fringes

An important property of incremental parsing is that for each partial derived tree, only a small part of the tree structure is available for substitution, adjunction and verification operations, as no operations before the last lexical anchor are possible, and any insertion of lexical material beyond the next substitution site or predicted beyond the lexical anchor would necessarily lead to a violation of incrementality later on, and not constitute a valid PLTAG partial derivation as defined in Section 7.2.1. We call the part of the tree that is available for substitution or adjunction the *current fringe*. It contains all nodes on the path from the last lexeme l_n to the tree's next leaf l_{n+1} (excluding their common ancestor). Paths from the leaf l_{n+1} to l_{n+2}, l_{n+2} to l_{n+3} etc. are referred to as the tree's *future fringes*. The last future fringe is the path from the rightmost leaf of the tree back to its root.

An elementary tree's current fringe is the path from its root node to the first leaf. We write a fringe as two lists, one list for nodes from the leaf on the way up that are open for adjunction to the right, and the second list for the nodes on the branch down to the next leaf with nodes open to the left for adjunction. Furthermore, we mark the existence of a substitution node, which is technically the last element of the second list,

with a colon at the end of the second list (or : *nil* if there is no substitution node). We can then also define the list of *past fringes* as those parts of the tree that were current fringes at previous points in time, i.e. paths between the root and the first leaf, the first leaf and second leaf etc. up to the path from the previous leaf to the current leaf. For example, consider the tree shown in Figure 8.14(a):

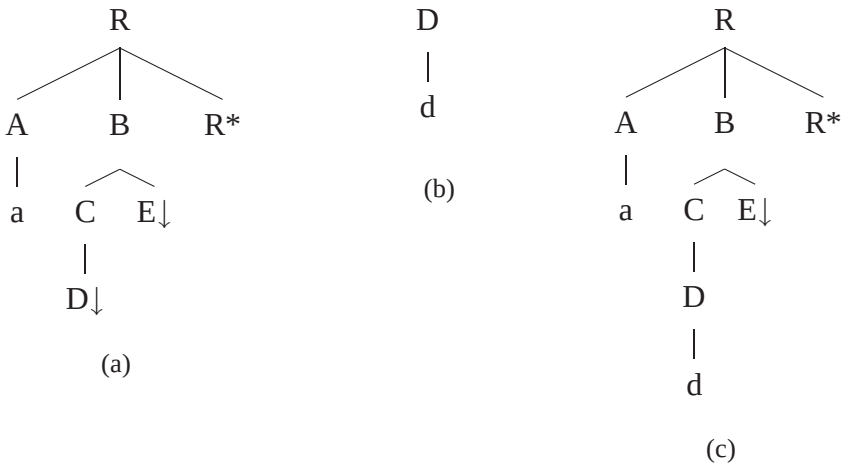


Figure 8.14: Example Trees for explaining the concept of fringes.

When first retrieving a tree from the lexicon, none of the leaves have been processed. Therefore the past fringe H is empty, and the current fringe F is $[][R,A,a] : nil$. The future fringes would be a list of fringes:
$P = [[a,A][B,C] : D\downarrow, \quad [D\downarrow,C][] : E\downarrow, \quad [E\downarrow, B][R*] : nil, \quad [R*,R][] : nil]$.
Once lexical item *a* has been processed, the formerly current fringe shifts onto the list of past fringes, ($H = [[][R,A,a] : nil]$), the first slice of the future fringes becomes the new current fringe ($C = [a,A][B,C] : D\downarrow$), and the future fringes consist of the remaining fringes ($P = [[D\downarrow,C][] : E\downarrow, \quad [E\downarrow,B][R*] : nil, \quad [R*,R][] : nil]$). If we then combine the tree achored in *a* with the tree shown in Figure 8.14(b) which has past fringe $H = []$ current fringe $C = [][D,d] : nil$ and future fringes $P = [[d,D][] : nil]$, we obtain the tree shown in Figure 8.14(c). The algorithm can efficiently decide whether trees (a) and (b) can be combined by checking their current fringes. The current fringe of tree (a) after processing non-terminal *a* ($C = [a,A][B,C] : D\downarrow$) contains a substitution node with the same category as the root category of tree (b), which is the first node on (b)'s current fringe $C = [][D,d] : nil$. When integrating trees (a) and (b), their fringes are combined such that they yield the fringe of tree (c). When tree (b) is substituted into node $D\downarrow$ of tree (a), their fringes must be *joined* to yield the fringe of tree (c).

Joining Fringes The fringe joining operation '+' appends the lists of the two fringes. This operation can of course only yield a valid fringe, if either the first list of the second fringe is empty, or if the second list and substitution node of the first fringe are empty, and if the node where they are joined, i.e. the last node of the first fringe and the first node of the second fringe have the same category. These conditions are satisfied in our example for the current fringes of trees (a) and (b) $[a,A][B,C] : D\downarrow + [][D,d] : nil$. Remember that the substitution node is technically part of the second list in a fringe. As substitution nodes are only the upper half of a complete node, and root nodes are only the lower half of a complete node, the second list of the second fringe is appended onto the second list of the first fringe, and the substitution node melted together with the root node, yielding fringe $[a,A][B,C,D,d] : nil$. The future fringes of trees (a) and (b) must also be joined: $[d,D][] : nil + [D\downarrow,C][] : E\downarrow$. In this case, the second list and substitution node of the frist fringe are empty, so the fringes satisfy requirements and the first list of the first fringe can be prepended to the second fringe, yielding resulting fringe $[d,D,C][] : E\downarrow$. Node halves for the root node and the substitution nodes are again melted together into a complete node D. We will refer back to this concept of joining fringes when discussing parser operations in Section 8.3.3.

When trees (a) and (b) from our example have been combined into tree (c) and non-terminal d has been processed, the resulting past fringe is thus $P = [][R,A,a] : nil, [a,A][B,C,D,d] : nil]$, the current fringe C is $[d,D,C][] : E\downarrow$ and future fringes consist of $P = [[E\downarrow,B][R*] : nil, [R*,R][] : nil]$. The operations in our incremental algorithm will make use of the fringe concept, and the rules specified in Section 8.3.3 take care of correctly putting together fringes when integrating two trees.

Note that if the past fringes, current fringe and future fringes of a tree are flattened and appended, we obtain the depth-first traversal order of a tree, also referred to as left-to-right tree traversal in the classical Tree-Adjoining Grammar paper in (Rozenberg and Salomaa, 1997, Section 10.1). For tree (c), this left-to-right traversal would correspond to the order $[R,A,a,a,A,B,C,D,d,d,D,C,E\downarrow,E\downarrow,B,R*,R*,R]$. Each node is visited exactly twice, once when integration can be performed on its left side, and once when operations can be applied at its right side (of course, no integrations can be applied at terminal symbols). Distinguishing between whether a node is visited for the first or second time is important for adjunction operations: an auxiliary tree with the foot as its rightmost child can only adjoin into a node that is open for adjunction from the left (i.e. on the second list of the fringe), while an auxiliary tree that has its foot node as its leftmost child can only integrate with nodes that are open

for adjunction to the right and hence in the first list of a fringe. Because a fringe is defined as the path from one leaf to the next, it always contains right-open nodes first (from the way up from the left leaf), and then left-open nodes (on the path down to the right leaf), which is why the chosen two-list-notation works well for displaying the nodes in the order they are visited by depth-first traversal. All operations maintain the correct order of nodes in the fringes, because all the components from a new tree that are inserted into the old fringes are themselves valid depth-first traversal orders, and they are inserted into the integration points where the trees are joined together, updating both the left-visible and right-visible parts in the resulting node order. We will show that the parser operation guarantee to correctly maintain fringes in Section 8.3.4.

8.3.2 A Parsing Example

For illustration of how the algorithm works, this section will go through an example sentence and discuss the necessary parsing operations. Consider Figure 8.15, which shows the start of an incremental PLTAG parse for the sentence *The reporter that the senator attacked, admitted the error*. When the first word, *the,* is encountered, there is nothing to integrate it with, so we apply the `Start` operation. The current fringe after the operation begins at the lexical anchor *the*. In the figure, it is indicated by the red dashed line and shown in the fringe notation that was discussed in the previous section.

Next, at the word *reporter*, the prefix tree from Figure 8.15(a) fits together with the new elementary tree (b) by performing a substitution operation (this is determined by checking the compatibility of the two trees' fringes. Because the prefix tree is substituted into the elementary tree (and not the elementary tree into the prefix tree) and is a canonical tree, this is a `SubstUpF` operation. The fringe of the resulting tree is given below the right hand side of Figure 8.15(b) and indicated in the tree by the red dashed line.

Next, we read the word *that* and, after checking fringes, find that the prefix tree from Figure 8.15(c) has an NP node that is open to the right, which means that the auxiliary tree for *that* (also Figure 8.15(c)) with its NP-foot node as the leftmost child can adjoin at this point. Since the auxiliary tree is a canonical tree, we apply operation `AdjDownF`. The resulting tree and fringe (again, indicated by the red dashed line) are given at the right-hand side of Figure 8.15(c). Note that adjunction of left-footed auxiliary trees has an effect on the past fringe. The

past fringe for the prefix tree was $[[][NP,DT,the] : nil, \quad [the,DT][NN,reporter] : nil]$. Adjoining to a right-open node means that we have to find the left side of the node in the former fringe and combine it with the current fringe of the tree which is being integrated. This yields the correct new past fringe $[[][NP,NP,DT,the] : nil, \quad [the,DT][NN,reporter] : nil]$. The fringe up to the node at which the integration took place $[reporter,NN,NP]$, is combined with the first future fringe of the elementary tree, yielding $[reporter,NN,NP][RC,WHNP,that] : nil$. (For the correct calculation of the derivation it is not necessary for the algorithm to maintain correct past fringes, since these will not be used any more. It is however crucial to maintain correct current and future fringes.)

When processing the word *the*, we find that the current fringes of the trees cannot be combined. Here's where the prediction trees come into play. The algorithm can take (and will try to do so after each input word) prediction trees from the prediction lexicon and try to integrate them into the prefix tree. One of the prediction trees in the set is the pre-combined prediction tree shown in Figure 8.15(d). Note the indices $_1$ and $_2$, which indicate which nodes should be verified by the same tree later. The prediction tree can be substituted into the open substitution node of the prefix tree (hence we use operation `SubstDownP`). But because the prediction tree does not have a lexical anchor, the current fringe is not shifted – instead, the fringes of the prediction tree replace the integration point in the prefix tree current fringe, resulting in the new current fringe $[that,WHNP][S_1,NP_2^1] : DT^2$, and future fringes $[[DT^2][NN_2^2] : nil, \quad [NN_2^2,NP_2^1][VP_1^1] : nil, \quad [VP_1^1,S_1,RC,NP][] : nil]$. Now, the current fringe of the new prefix tree actually contains a DT-substitution node and can thus be combined with the *the*-tree, using operation `SubstDownF`. After fringe combination, the fringe is shifted to after the last lexical anchor, "the", as shown in Figure 8.15(e), indicated by the red dashed line. The current fringe of the prefix tree now ends with a prediction node, and is therefore a candidate for verification (we cannot shift fringes before this predicted node is verified – we could however integrate other words into the current fringe before verification).

For the following word, *senator*, there exists a canonical tree in the lexicon that can verify all predicted nodes with index $_2$. The indices of the validated nodes on the fringes are removed, and the fringes are updated where necessary (i.e. wherever the verification tree contains nodes that are not contained in the prediction tree). This is done by replacing the leaves of the prediction tree by the corresponding nodes in the verification tree. So the current fringe is updated from $[the,DT^2][NN_2^2] : nil$ to

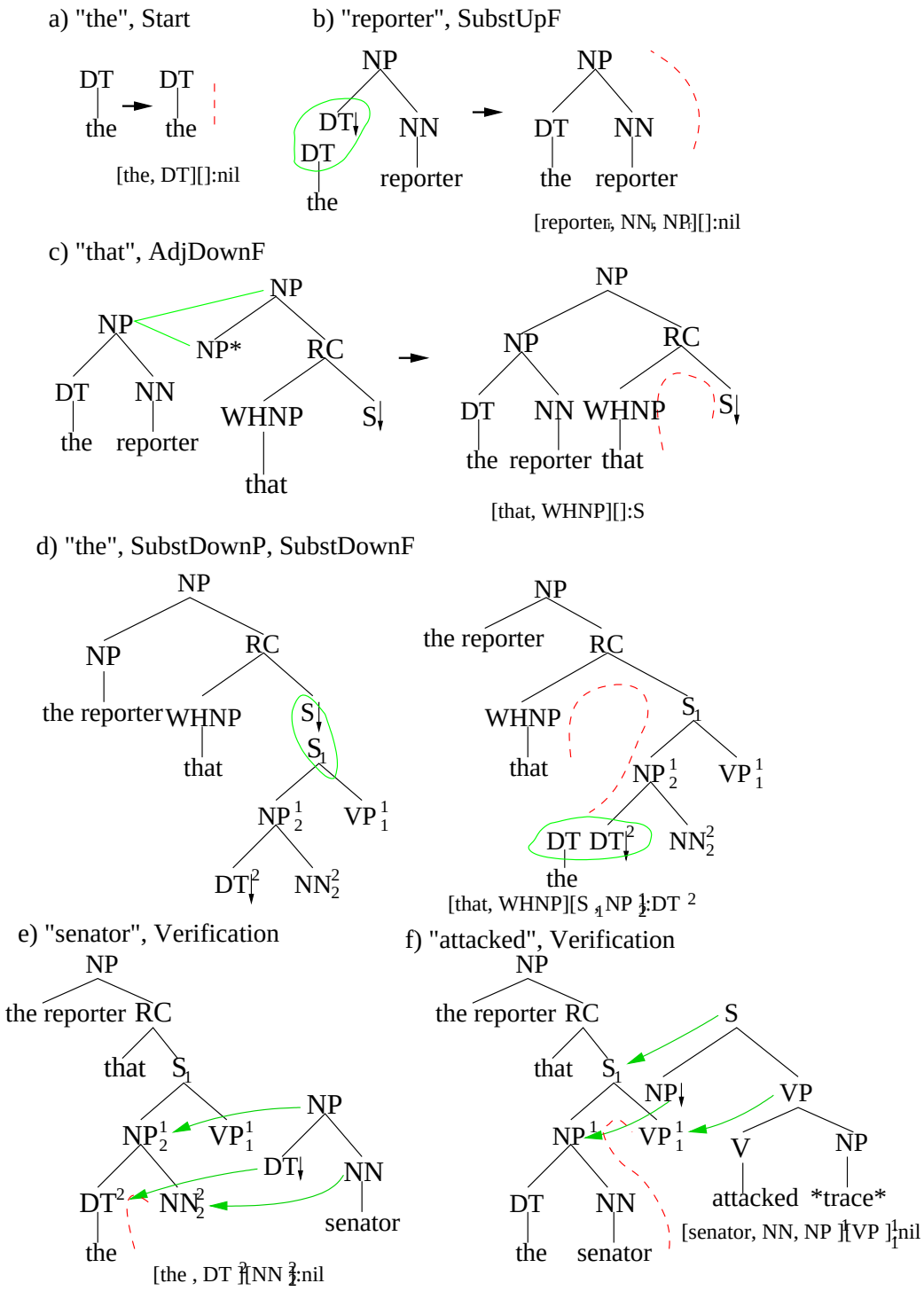


Figure 8.15: Incrementally parsing an object relative clause with PLTAG.

$[the, DT][NN, senator] : nil$ and the first slice of the future fringe is changed from $[NN_2^2, NP_2^1][VP_1^1] : nil$ to $[senator, NN, NP^1, VP_1^1] : nil$. The next word, *senator*, has an elementary tree that matches the predicted nodes with index $_1$, see Figure 8.15(f). The future fringe resulting from the verification operation starts from the lexical an-

chor *attacked*, yielding current fringe $[attacked, V][NP, *trace*] : nil$ and future fringe $[[*trace*, NP, VP, S, RC, NP][] : nil]$. Note that it would be grammatical (and indeed, quite likely in this case) to next encounter a word that integrates with a position on the future fringe, after the trace. Because the trace is an empty element, both fringes should be available for integrating the next word. Therefore, a second analysis is created, which is an exact copy of the first but whose current fringe starts after the trace, yielding current fringe $[*trace*, NP, VP, S, RC, NP][] : nil$ and an empty future fringe.

8.3.3 Formalisation of the Parser Operations

The parser can choose between five parser operations (`SubstDownF`, `SubstUpF`, `AdjDownF`, `AdjUpF`, `Verification`) for combining a prefix tree with a canonical elementary tree, and four operations for combining a prefix tree with a prediction tree (`SubstDownP`, `SubstUpP`, `AdjDownP`, `AdjUpP`). In addition, there is the `Start` operation which processes the tree for the first word in the sentence. Taken together, these operations implement the PLTAG grammar operations (adjunction, substitution, and verification) as operations on fringes. For efficiency reasons, we only keep track of the trees' fringes during search, and build the derived trees during retrieval of the *n*-best analyses. The role of the parsing operations is to guarantee that valid prefix trees are generated by checking all preconditions are satisfied before integrating a tree, and to correctly calculate the current and future fringes of the resulting trees. The input of the parser operations are the fringes of the prefix tree and elementary tree, and a list of the prefix trees that have been previously integrated into the prefix tree, but have not yet been verified. This list is necessary as some of the nodes of the prediction tree may not be present on the current or future fringe at the time of verification, but are needed to check whether the prediction and verification tree match.

The following paragraphs describe the parser operations in more detail. The prefix tree β and the elementary tree ε can each be represented as tuples T and T', see Table 8.2.

A parser operation takes the two tuples for the trees[2] and generates the tuple for the resulting tree as spelled out below. All operations will be explained by listing their preconditions for integrating two trees, and their calculation of the resulting tree's current and future fringes.

[2] In fact, just the fringes and the position of the word that's being processed are enough as C and X can be determined automatically given the trees' fringes.

Tuple for representing parser operations:	
C	the category of the tree's root node
X	indicates the position of a tree's foot node. Values are 'r' (rightmost leaf), 'l' (leftmost leaf) and '-' (no foot node for initial trees)
i	index of last subsumed word, '-' for elementary prediction tree
F	a tree's current fringe, can also be expressed as $[\underset{1..m}{A}][\underset{1..o}{B}]$:S, see Section 8.3.1
$\underset{1..n}{P}$	The tree's *n* future fringes.
Other definitions:	
$\overset{k}{k}$	the index *k* as a subscript or superscript marks predicted nodes.
L	denotes the last canonical anchor of a tree, if it is part of a fringe, always last node on second list in a fringe
N	a node
$T \bowtie T'$	tree T is integrated with tree T'
$\oplus$	merge fringes in correct order
+	join fringes, see Section 8.3.1

Table 8.2: Definitions for the specifications of parsing operations.

Start

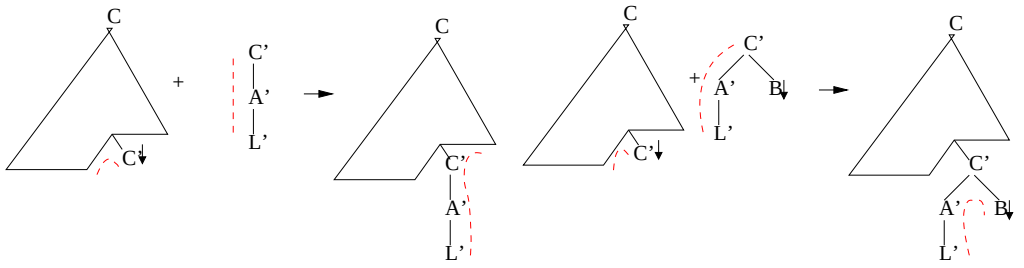
$$\frac{[] \;\bowtie\; (C;\, X = - \vee r;\, 1;\, [][\underset{1..o-1}{B}, L] : nil;\, \underset{1..n}{P})}{(C;\, X;\, 1;\, \underset{1}{P};\, \underset{2..n}{P})}$$

Preconditions of the **Start** operation are that the elementary tree must not be a prediction tree. It can be either an initial tree, or an auxiliary tree with its foot to the right (expressed as $X = - \vee r$ in the formula), and it must not have an open substitution node to the left of the lexical anchor. To check these conditions, the operation must verify that the elementary tree's current fringe ends with a lexical anchor L. Finally, this condition can only apply if there is no prefix tree and the number of the word processed is 1. The operation then sets the new current fringe to the first future fringe of the elementary tree P_1, and future fringes to $P_{2\ldots n}$.

SubstDownF

$$\frac{(C;\, X;\, 1..i;\, [\underset{1..m}{A}][\underset{1..o}{B}] : S;\, \underset{1..n}{P}) \;\bowtie\; (C';\, -;\, i+1;\, [][\underset{1..o'-1}{B'}, L'] : nil;\, \underset{1}{P'})}{(C;\, X;\, 1..i+1;\, \underset{1}{P'} + \underset{1}{P};\, \underset{2}{P} \ldots \underset{n}{P})} \; cat(S) = C'$$

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}]:S;\ \underset{1..n}{P})\ \bowtie\ (C';\ -;\ i+1;\ [][\underset{1..o'-1}{B'},L']:nil;\ \underset{|P'|>1}{P'})}{(C;\ X;\ 1..i+1;\ \underset{1}{P'};\ \underset{2..n'-1}{P'},\underset{n'}{P'}+\underset{1}{P},\ \underset{2..n}{P})}\ cat(S)=C'$$



(a) initial tree with $|P| = 1$ (b) initial tree with $|P| > 1$

Figure 8.16: Illustration of the **SubstDownF** operation. The red dashed lines indicate the current fringe. The names of the nodes in the figure do not correspond directly to the variable names in the formulae. This also holds for the other figures in this section. The S node in the right hand side of the formula is depicted as C' in the figure (following the equivalence condition *cat(S) = C'*.

Preconditions for the **SubstDownF** operation are that the elementary tree must be an initial tree, therefore there is no foot node (marked as '$-$' in the formula). For the prefix tree, it does not matter whether it is an auxiliary tree or not: The X marks its status as underspecified). Furthermore, the prefix tree's substitution node S must have the same category as the root node of the elementary tree C'. In order not to violate incrementality, the first leaf of the elementary tree must be the lexical anchor (indicated in the formula by the L' on the current fringe).

To correctly calculate the current and future fringes of the resulting tree, two cases are distinguished: the future fringe of the elementary tree contains has length 1 ($\underset{1}{P'}$) or length > 1 ($\underset{|P'|>1}{P'}$). If the length of the future fringe is one, the current fringe of the resulting tree is made up of the future fringe slice of the elementary tree, joined at the substitution site with the first future fringe of the prefix tree. Otherwise, the current fringe of the resulting tree is simply equal to the first future fringe of the elementary tree, see Figures 8.16 (a) and (b). In both cases, the last part of the future fringe of the elementary tree needs to be joined with the first slice of the future fringe of the prefix tree, as indicated by the '+', see Section 8.3.1 for an explanation for how fringes are joined.

SubstDownP

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}] : S;\ \underset{1..n}{P}) \ \bowtie\ (C';\ -;\ -;\ [][\underset{1..o'}{B'^{k}_{k}}] : S'^{k};\ \underset{1..n'}{P'^{k}_{k}})}{(C;\ X;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}] : S + [][\underset{1..o'}{B'^{k}_{k}}] : S'^{k};\ \underset{1..n'-1}{P'^{k}_{k}}, P'^{k}_{n'} + \underset{1}{P}, \underset{2..n}{P})}\ cat(S) = C'$$

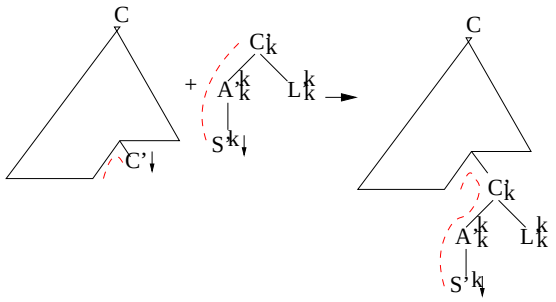


Figure 8.17: Illustration of the **SubstDownP** operation. The red dashed lines indicate the currently accessible fringe.

Preconditions for the **SubstDownP** operation are that the elementary tree must be an initial tree, indicated by '–' in second position of the elementary tree tuple. Prediction trees are not lexicalized. Therefore, we do not account for any additional words in the input string (as shown by the '$-$' in the third position of the elementary tree tuple). The indices at nodes in the elementary tree make the required prediction status of the elementary tree for the **SubstDownP** operation explicit. The inner nodes have both an upper and a lower index and the substitution nodes only have upper indices. Again, the root node of the elementary tree must have the same category as the substitution node S on the current fringe of the prefix tree. As opposed to the *SubstDownF* operation, the elementary tree is allowed to have an open substitution node as a first leaf, as indicated by the disjunction 'S'^{k}').

The fringe of the resulting tree is made up out of the current fringe of the prefix tree, joined with the current fringe of the elementary tree. The future fringes consist of the first $n'-1$ future fringes of the elementary tree $\underset{1..n'-1}{P'^{k}_{k}}$, then followed by the elementary tree's last future fringe which is joined with the prefix tree's first future fringe $P'^{k}_{n'} + \underset{1}{P}$. The last part of the resulting tree's future fringe is the rest of the prefix tree's future fringes $\underset{2..n}{P}$. Down-substitution with prediction trees is illustrated in Figure 8.17.

SubstUpF

$$\frac{(C;\ -;\ 1..i;\ [\underset{1..m}{A}][] : nil;\ []) \ \bowtie\ (C';\ X';\ i+1;\ [][\underset{1..o'}{B'}] : S';\ [\underset{1..n''}{A''}][\underset{1..o''}{B''}, L''] : nil, \underset{2..n'}{P'})}{(C';\ X';\ 1..i+1;\ \underset{2}{P'};\ \underset{3..n}{P'})}\ cat(S') = C$$

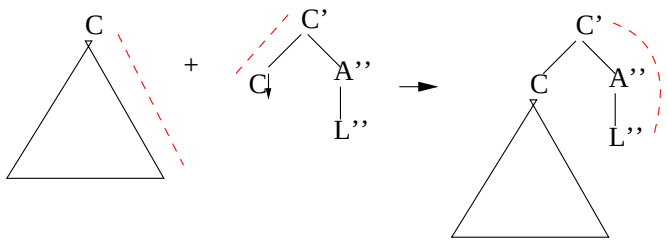


Figure 8.18: Illustration of the **SubstUpF** operation. The right branch of the elementary tree might of course be much more complex than the one of the example tree shown here. P'_1 is spelled out here as $[\,A''_{1..n''}][\,B''_{1..o''}, L'']:nil$.

Preconditions of the **SubstUpF** operation are that the future fringes of the prefix tree must be empty. The elementary tree must have the substitution node as a first leaf, which matches the root node of the prefix tree, and a lexical anchor as the second leaf. To make this clear in the formula, P'_1 is spelled out as $[\,A''_{1..n''}][\,B''_{1..o''}, L'']:nil$. After combination of the trees, the current fringe starts at the lexical anchor of the elementary tree, see Figure 8.18.

SubstUpP

$$\frac{(C;\ -;\ 1..i;\ [\,A_{1..m}\,][]:nil;\ [])\ \bowtie\ (C';\ X';\ -;\ [][\,B'^{\,k}_{1..o'\,k}]:S'^k;\ P'^{\,k}_{1..n'\,k})}{(C';\ X';\ 1..i;\ [\,A_{1..m}\,][]:nil+P'^{\,k}_{1\,k};\ P'^{\,k}_{2..n'\,k})}\ cat(S'^k)=C$$

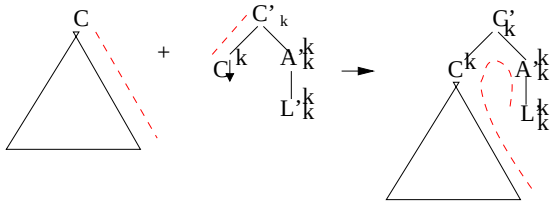


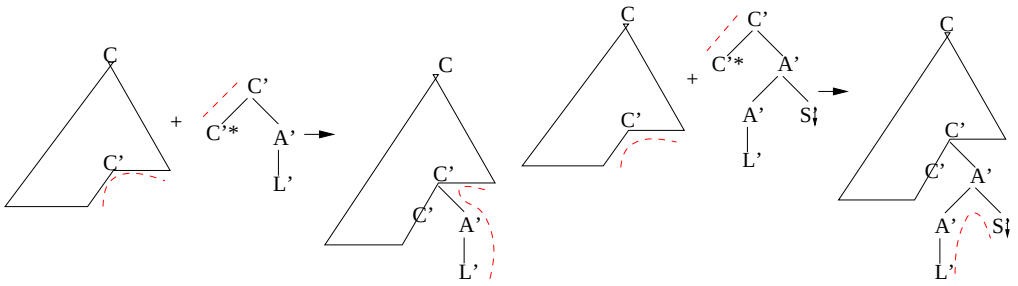
Figure 8.19: Illustration of the **SubstUpP** operation.

The preconditions for this operation are, again, that the future fringes of the prefix tree are empty, that the elementary tree is a prediction tree (as indicated by the ^k_k indices on its nodes) and that it has a substitution node on its current fringe with the same category as the prefix tree's root node. In a substitution, the root node of the elementary tree and the substitution node of the prefix tree are merged into a single node, and the future fringe must thus be updated by joining the prefix tree's current fringe $[\,A_{1..m}\,][]:nil;$ with the elementary tree's first future fringe $P'^{\,k}_{1\,k}$. The future fringes of the resulting tree are the future fringes of the elementary tree $P'^{\,k}_{2..n'\,k}$, see Figure 8.19.

AdjDownF (foot in auxiliary tree to the left)

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..j-1}{A},\underset{j}{A},\underset{j+1..m}{A}][\underset{1..o}{B}]:S;\ \underset{1..n}{P})\ \bowtie\ (C';\ l;\ i+1;\ [][\underset{1..o'}{B'},C'*]:nil;\ \underset{1..2}{P'})}{(C;\ X;\ 1..i+1;\ \underset{2}{P'}+[\underset{j+1..m}{A}][\underset{1..o}{B}]:S;\ \underset{1..n}{P})}\ cat(\underset{j}{A})=C'$$

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..j-1}{A},\underset{j}{A},\underset{j+1..m}{A}][\underset{1..o}{B}]:S;\ \underset{1..n}{P})\ \bowtie\ (C';\ l;\ i+1;\ [][\underset{1..o'}{B'},C'*]:nil;\ \underset{|P'|>2}{P'})}{(C;\ X;\ 1..i+1;\ \underset{2}{P'};\ \underset{3..n'-1}{P'},\underset{n'}{P'}+[\underset{j+1..m}{A}][\underset{1..o}{B}]:S,\underset{1..n}{P})}\ cat(\underset{j}{A})=C'$$



(a) auxiliary tree with $|P|=2$ (b) auxiliary tree with $|P|>2$

Figure 8.20: Illustration of the **AdjDownF** operation for auxiliary trees with the foot as their first leaf. The red dashed lines indicate the current fringe. Note that the C' in the prefix trees corresponds to the $\underset{j}{A}$ in the formula, as indicated by $cat(\underset{j}{A})=C'$.

(foot in auxiliary tree to the right)

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..n}{A}][\underset{1..j-1}{B},\underset{j}{B},\underset{j+1..o}{B}]:S;\ \underset{1..n}{P})\ \bowtie\ (C';\ r;\ i+1;\ [][\underset{1..o'}{B'},L']:nil;\ \underset{1..2}{P'})}{(C;\ X;\ 1..i+1;\ \underset{1}{P'}+[\underset{j+1..o}{B}]:S;\ \underset{1..n}{P}\oplus\underset{2}{P'})}\ cat(\underset{j}{B})=C'$$

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..n}{A}][\underset{1..j-1}{B},\underset{j}{B},\underset{j+1..o}{B}]:S;\ \underset{1..n}{P})\ \bowtie\ (C';\ r;\ i+1;\ [][\underset{1..o'}{B'},L']:nil;\ \underset{|P'|>2}{P'})}{(C;\ X;\ 1..i+1;\ \underset{1}{P'};\ \underset{2..n'-2}{P'},\underset{n'-1}{P'}+[\underset{j+1..o}{B}]:S,\underset{1..n}{P}\oplus\underset{n'}{P'})}\ cat(\underset{j}{B})=C'$$

The preconditions for applying the **AdjDownF** operation are that the elementary tree is a canonical auxiliary tree with no substitution site before its anchor, and that the fringe of the prefix tree contains a possible adjunction site for the auxiliary tree. The function $cat(\underset{j}{A})=C'$ means that the category of the adjunction site $\underset{j}{A}$ has to match the category of the elementary tree's root node. Note that we do only consider TIG auxiliary trees here (i.e. trees whose foot node is not the left- or right-most child).

The fringes of the resulting tree are calculated as follows. Consider first the case where the foot node is the leftmost leaf of the auxiliary tree. The current fringe of the

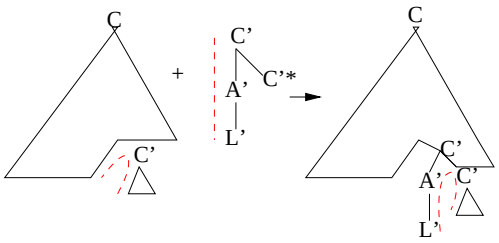


Figure 8.21: Illustration of the **AdjDownF** operation for auxiliary trees with the foot as their rightmost leaf.

resulting tree is the second future fringe of the auxiliary tree (starting at the lexical anchor). If there are no further future fringes, the elementary tree's current fringe is joined with the prefix tree's current fringe after the adjunction site, and the future fringes of the resulting tree are simply the same as the future fringes of the prefix tree. If the auxiliary tree has more than two future fringes, the resulting current fringe is the elementary tree's second future fringe and the future fringes of the resulting tree start with the future fringes of the auxiliary tree $\underset{3..n'-1}{P'}$. The last future fringe is joined with the prefix tree's current fringe after the adjunction site $\underset{n'}{P'}+[\underset{j+1..m}{A}][\underset{1..o}{B}]:S$. The rest of the resulting tree's future fringe is made up of the future fringe of the prefix tree $\underset{1..n}{P}$.

When the foot node is the rightmost leaf of the auxiliary tree, the calculation of the resulting tree's future fringe is slightly more complicated. The current fringe of the resulting tree starts with the elementary tree's future fringes $\underset{1}{P'}$. If this fringe ends with the foot node (in the $\underset{|P|=2}{P'}$ case), the resulting tree's current fringe continues with the nodes after the integration node $\underset{j}{B}$. Otherwise the future fringes (up to the second last one) of the auxiliary tree constitute the future fringes of the resulting tree. The second last fringe is joined with the prefix tree's current fringe after the integration node.

The adjunction operation inserts the nodes on the path from the foot node to the root node of the auxiliary tree at the adjunction site $\underset{j}{B}$ in the resulting tree, see Figure 8.21. Because each node occurs exactly twice in the fringes of a tree, first as open to the left and later as open to the right (as depth-first tree traversal visits each node twice), the prefix tree's future fringe that contains the adjunction site node open to the right must be updated by inserting the last fringe of the auxiliary tree. This is noted in the formula as $\underset{1..n}{P} \oplus \underset{n'}{P'}$.

AdjDownP (auxiliary tree foot left)

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..j-1}{A}, \underset{j}{A}, \underset{j+1..m}{A}][\underset{1..o}{B}] : S;\ \underset{1..n}{P}) \bowtie (C';\ l;\ i+1;\ [][\underset{1..o'-1}{B'}\ \overset{k}{k}, C'^{k}*] : nil;\ \underset{1..n'}{P'}\ \overset{k}{k})}{(C;\ X;\ 1..i;\ [\underset{1..j-1}{A}, \underset{j}{A}] + \underset{1}{P'^{k}}{}_{k};\ \underset{2..n'-1}{P'}\ \overset{k}{k}, \underset{n'}{P'}\ \overset{k}{k} + [\underset{j+1..m}{A}][\underset{1..o}{B}] : S, \underset{1..n}{P})}\ cat(\underset{j}{A}) = C'$$

(auxiliary tree foot right)

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..n}{A}][\underset{1..j-1}{B}, \underset{j}{B}, \underset{j+1..o}{B}] : S;\ \underset{1..n}{P}) \bowtie (C';\ r;\ i+1;\ [][\underset{1..o'}{B'}\ \overset{k}{k}] : S'^{k};\ \underset{1..n'}{P'}\ \overset{k}{k})}{(C;\ X;\ 1..i+1;\ [\underset{1..n}{A}][\underset{1..j-1}{B}, \underset{j}{B}] + [][\underset{1..o'}{B'}\ \overset{k}{k}] : nil;\ \underset{1..n'-2}{P'}\ \overset{k}{k}, \underset{n'-1}{P'} + [\underset{j+1..o}{B}] : S, \underset{1..n}{P} \oplus \underset{n'}{P'}\ \overset{k}{k})}\ cat(\underset{j}{B}) = C'$$

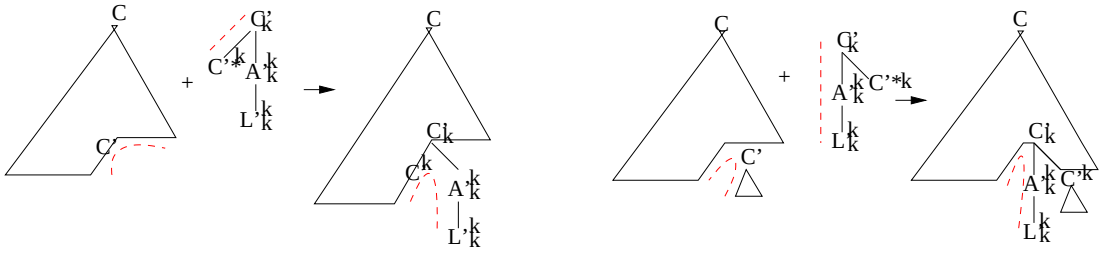


Figure 8.22: Illustration of the **AdjDownP** operation.

The **AdjDownP** operation works the same as the **AdjDownF** operations, except the auxiliary tree must be a prediction tree, and may have substitution nodes before the anchor. When calculating the current and future fringes of the resulting tree, the current fringe starts at the current fringe of the prefix tree, i.e. it is not shifted to after the anchor of the auxiliary tree, as no lexeme was processed. An example of the operation is shown in Figure 8.22).

AdjUpF

$$\frac{(C;\ r;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o-1}{B}, C*] : nil;\ \underset{1}{P}) \bowtie (C';\ X' = r \vee -;\ i+1;\ [][\underset{1..j-1}{B'}, \underset{j}{B'}, \underset{j+1..o'}{B'}, L'] : nil;\ \underset{1..n'}{P'})}{(C';\ X';\ 1..i+1;\ \underset{1}{P} \oplus \underset{1}{P'};\ \underset{2..n'}{P'})}\ cat(\underset{j}{B'}) = C$$

Preconditions for applying the **AdjUpF** operation are that the prefix tree must be an auxiliary tree with the foot node as its rightmost leaf, and that the length of the future fringe must equal one. Furthermore, the category of the prefix tree's foot and root node must be compatible with a node on the elementary tree. The elementary tree must be a canonical tree, and must not have any leaf to the left of its lexical anchor. It may itself be an auxiliary tree (with foot to the right).

When the prefix tree is adjoined into the elementary tree, the current fringe of the resulting tree is the first future fringe of the elementary tree. The current fringe of the resulting tree however also contains the elementary tree's fringe after the adjunction

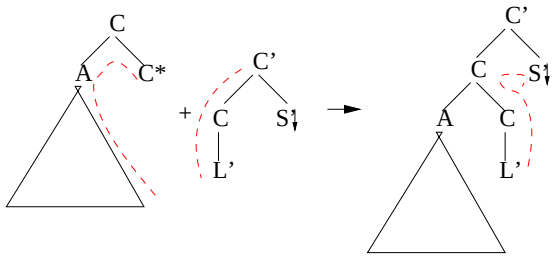


Figure 8.23: Illustration of the **AdjUpF** operation. The red dashed lines indicate the current fringe.

site. Note that the future fringe of the prefix tree contains the path from the foot node to the root node. This fringe needs to be joined with the future fringe of the elementary tree at the adjunction site (for nodes accessible from the right). See this effect in Figure 8.24, where the first slice of the future fringe of the elementary tree only contains two C nodes, while the future fringe of the resulting tree after the operation contains three C-nodes. The future fringe of the resulting tree is the same as the rest of the future fringe $\underset{2..n'}{P'}$ of the elementary tree.

AdjUpP

$$\frac{(C;\ r;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o-1}{B}, C*] : nil;\ \underset{1}{P}) \bowtie (C';\ X';\ -;\ [][\underset{1..j-1}{B'}{}^{k}_{k}, B'^{k}_{j}, \underset{j+1..o'}{B'}{}^{k}_{k}] : S^k;\ \underset{1..n'}{P'}{}^{k}_{k})}{(C';\ X';\ 1..i;\ [\underset{1..m}{A}][\underset{1..o-1}{B}, C] : nil + [\underset{j+1..o'}{B'}{}^{k}_{k}] : S^k;\ \underset{1}{P} \oplus \underset{1..n'}{P'}{}^{k}_{k})} cat(B'^{k}_{j}) = C$$

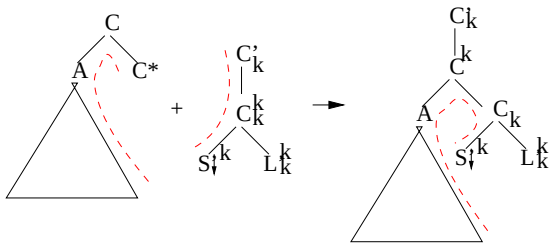


Figure 8.24: Illustration of the AdjUpP operation.

The **AdjUpP** operation is very similar to the **AdjUpF** operation, the only difference being that the prediction status of nodes needs to be taken care of, and that the leaves before the anchor can be substitution nodes. As in the other operations with a prediction tree, the fringe does not have to move beyond any anchor.

Verification

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}, L_k^k] : nil;\ \underset{1..n}{P}) \bowtie (C';\ X';\ i+1;\ [][\underset{1..o'}{B'}] : S';\ \underset{1..i'-1}{P'}, [\underset{1..m''}{A''}][\underset{1..o''}{B''}, L''] : nil, \underset{i'+1..n'}{P'})}{(C;\ X;\ 1..i+1; \underset{1}{P} \oplus \underset{i'+1}{P'};\ \underset{1}{P} \oplus \underset{i'+2..n}{P'}, \underset{2..n}{P})} m(\varepsilon, \underset{all}{N_k^k})$$

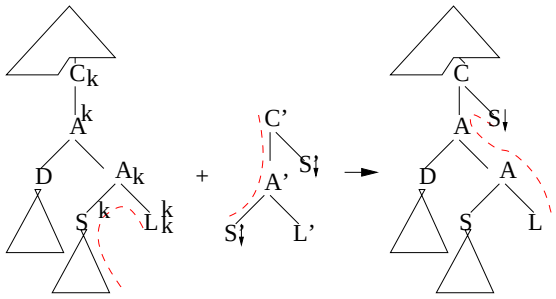


Figure 8.25: Illustration of the **Verification** operation. The red dashed lines mark the current fringe.

Verification is the only operation during which the prediction status of a node can be changed. It can only be applied if the last node of the fringe (i.e. the upcoming leaf) is marked as a predicted node (if there was an open substitution node, it would have to be filled first in order not to violate incrementality later on). There exist no "up-" and "down-" versions of this operation because verification is only triggered by a constellation where the prefix tree contains a configuration of prediction nodes that is compatible with the node configuration of a canonical elementary tree ε (we denote this using the function $m(\varepsilon, \underset{all}{N_k^k})$). The **Verification** operation removes prediction markers ^{k} and $_k$ from all nodes that are validated. This can of course affect the prediction statuses of nodes in the past, current and future fringes (the above formula does not explicitly show the prediction status changes on the fringes; this is hidden in the $\oplus$ operation applied to the future fringes of the prefix and verification trees). The $\oplus$ symbol means that the future fringes are merged together correctly by replacing prediction nodes with canonical nodes in the correct order, adding any additional nodes (i.e. substitution nodes to the right of the spine which are part of the verification tree but were not part of the prediction tree), and keeping track of the adjunctions that were made to the prediction spine before, as well as adding any remaining nodes of the prefix tree's future fringes. The reason for the $\underset{1}{P}$ operator showing up twice is due to the fact that if the right side of the spine contains any nodes that are not part of the original prediction tree, there will be additional fringe slices. Therefore, only part of the original $\underset{1}{P}$ fringe will be part of the current fringe, and part of it will be pushed to the future fringe. For an example of the verification operation, see Figure 8.25.

Dealing with Traces and Empty Elements One exception to the choice of where the current fringe starts occurs at traces and empty elements. After applying all operations as usual, the resulting fringes are checked to determine whether the next leaf (the last node on the current fringe) is a trace or null-element. If that's the case, the analysis is copied and in the copy, the current fringe is shifted to the first future fringe.

8.3.4 Proof that Operations produce only valid PLTAG Derivations

This section demonstrates that the operations specified above produce valid PLTAG derivations by showing that they are designed to satisfy the following start and end conditions, as well as invariants for partial derivations, which are taken from the PLTAG derivation definition from Section 7.2.1.

- Start Condition for Derivation
  At the start of a derivation, PLTAG only allows the use of canonical trees, but no prediction trees. Since the lexicon of canonical trees in PLTAG and LTAG are the same, the first tree must be a valid partial LTAG derivation. PLTAG in addition requires that the initial tree must not have any open substitution trees to the left of its anchor, and that, if it is an auxiliary tree, the foot must be right of its spine.

- Invariants that hold for partial derivations

  1. The current fringe of a derived tree is always the path between the last processed word and the next leaf.
  2. If the past fringe, the current fringe and the future fringe are flattened and appended, the order of nodes corresponds to the depth-first search path of the partial derivation.
  3. A partial derivation tree that covers word w_j as the most recently processed word also covers words $w_1..w_j$.
  4. In a partial derivation, all leaves before word w_j are lexicalized (i.e. canonical, not feet, and not substitution nodes).
  5. For all prediction markers with the same index, their number must either remain constant after introduction, or be equal to 0 (after verification).

- End Condition for Derivation

  At the end of a valid PLTAG derivation, there must not be any nodes marked as predicted, and no open substitution nodes or foot nodes.

How do the rules maintain these conditions?

We here discuss the rules presented above with respect to how they guarantee yielding valid PLTAG derivations. All tables contain three rows, one for the prefix tree β, one for the elementary tree ε and one for the resulting tree r.

Start

The **Start** rule (repeated here for convenience of reference), only allows canonical trees. It requires the first leaf to be a lexeme (L), and the foot node, if it exists, to be the rightmost leaf. It therefore satisfies the start condition as stated above. Furthermore, the flattened and appended fringes of the initial tree must by definition be in depth-first search order. This operation maintains this order, and thus satisfies invariant 2.

$$\frac{[] \bowtie (C;\, X = - \vee r;\, 1;\, [][\underset{1..o-1}{B}, L] : nil;\, \underset{1..n}{P})}{(C;\, X;\, 1;\, \underset{1}{P};\, \underset{2..n}{P})}$$

	past fringe	current fringe	future fringe
β			
ε	-	$[][\underset{1..o-1}{B}, L] : nil$	$\underset{1..n}{P}$
r	$[][\underset{1..o-1}{B}, L] : nil$	$\underset{1}{P}$	$\underset{2..n}{P}$

The new prefix tree r resulting from the operation covers exactly word w_1 (satisfying invariant 3). The current fringe is shifted to the path after the first lexical anchor, hence invariant 1 is also satisfied. Trivially, due to the start condition, there are no open substitution nodes or predicted nodes before leaf w_1 (invariant 4); invariant 5 also holds since no prediction trees have been used so far.

SubstDownF

The **SubstDownF** operations handle the substitution of a canonical tree into a matching substitution node in the prefix tree. (Here, the process is shown for $\underset{|P'|>1}{P'}$, but the same argumentation can be easily adapted to the case $\underset{|P'|=1}{P'}$.)

SubstDownF satisfies the invariants by maintaining the fringe condition (invariant 2): The prefix tree's flattened and appended fringes are in the same order as the depth-first search of the tree, and the same is true with respect to the elementary tree. The operation combines these fringes such that the resulting fringe also maintains the correct order: The nodes on the resulting fringe which originates from the elementary tree are in the same order as before, and they are inserted into the prefix tree at the integration point. The '+' operation combines nodes S (which only has an upper half) and node A'_1 (the root of the elementary tree which only has a lower half) into one single node. The same happens at the point when the integration point node occurs for the second time in the fringe, where the last node of the fringe slice $P'_{n'}$, which is the right hand side of the elementary tree's root node, is combined with the first node in P_1, the right hand side of the prefix tree's substitution node.

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}]:S;\ \underset{1..n}{P}) \ \bowtie\ (C';\ -;\ i+1;\ [][\underset{1..o'-1}{B'},L']:nil;\ \underset{|P'|>1}{P'})}{(C;\ X;\ 1..i+1;\ \underset{1}{P'};\ \underset{2..n'-1}{P'},\underset{n'}{P'}+\underset{1}{P},\ \underset{2..n}{P})}\ cat(S)=C'$$

	past fringe	current fringe	future fringe
β	$\underset{1..n}{H}$	$[\underset{1..m}{A}][\underset{1..o}{B}]:S$	$\underset{1..n}{P}$
ε	$-$	$[][\underset{1..o'-1}{B'},L']:nil$	$\underset{1..n'}{P'}$
r	$\underset{1..n}{H},[\underset{1..m}{A}][\underset{1..o}{B}]:S+[][\underset{1..o'-1}{B'},L']:nil$ $=\underset{1..n}{H},[\underset{1..m}{A}][\underset{1..o}{B},S+\underset{1}{B'},\underset{2..o'-1}{B'},L']:nil$	$\underset{1}{P'}$	$\underset{2..n'-1}{P'},\underset{n'}{P'}+\underset{1}{P},\ \underset{2..n}{P}$

The prefix tree covers words $w_1..w_i$. It does not have any open substitution nodes or predicted leaves before the leaf with word w_i, and the leaf following this node is the open substitution node S. During the operation, this open substitution node is filled, and we know from the elementary tree that this tree's first leaf contains the lexical anchor for w_{i+1}. Therefore, the operation guarantees that no substitution node or predicted leaf can exist between w_i and w_{i+1}, and thus satisfies invariants 3 and 4. The resulting current fringe is P'_1, which we know to be the path from w_{i+1} to the next leaf, thus satisfying invariant 2. The operation does not change anything in the prediction index annotation of the nodes of the two trees, hence invariant 5 is also valid after the operation.

SubstDownP

Next let us consider a substitution operation which introduces a prediction tree. Again,

we assume that the prefix and elementary trees have fringes in correct depth-first order, and themselves satisfy the invariants. As in the **SubstDownF** operation, the nodes remain in the same order and are inserted into the prefix tree fringe at the integration point, thus maintaining invariant 2. Because the elementary tree is a prediction tree, the current fringe is not shifted and thus maintains the prefix tree's properties of satisfying invariants 3 and 4. The current fringe of the resulting tree starts at the last processed word, and due to the update of the fringe with the elementary tree fringes, it contains the path from that word to the next leaf, thus satisfying invariant 1. All nodes introduced by the elementary tree contain indices, and these indices are not modified by the operation, thus maintaining invariant 5.

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}] : S;\ \underset{1..n}{P}) \ \bowtie\ (C';\ -;\ -;\ [][\underset{1..o'}{B'^k_k}] : S'^k;\ \underset{1..n'}{P'^k_k})}{(C;\ X;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}] : S + [][\underset{1..o'}{B'^k_k}] : S'^k;\ \underset{1..n'-1}{P'^k_k}, P'^k_{n'} + \underset{1}{P}, \underset{2..n}{P})}\ cat(S) = C'$$

	past fringe	current fringe	future fringe
β	$\underset{1..n}{H}$	$[\underset{1..m}{A}][\underset{1..o}{B}] : S$	$\underset{1..n}{P}$
ε	$-$	$[][\underset{1..o'}{B'^k_k}] : S'^k$	$\underset{1..n'}{P'^k_k}$
r	$\underset{1..n}{H}$	$[\underset{1..m}{A}][\underset{1..o}{B}] : S + [][\underset{1..o'}{B'^k_k}] : S'^k$	$\underset{1..n'-1}{P'}, P'_{n'} + \underset{1}{P}, \underset{2..n}{P}$
		$[\underset{1..m}{A}][\underset{1..o}{B}, S + \underset{1}{B'}, \underset{2..o'}{B'^k_k}] : S'^k$	

SubstUpF

In up-operations, the prefix tree is integrated into the elementary tree. Therefore the fringe from the prefix tree is inserted into the elementary tree fringe at the integration point. The past fringe now begins with the elementary tree's current fringe, joined at the substitution node with the first past fringe slice of the prefix tree, and followed by the remaining past fringe slices from the prefix tree. Then, the current fringe of the prefix tree is merged with the first slice of the elementary tree's future fringe. All nodes remain in order (see table below), and invariant 2 is thus satisfied. Invariants 3 and 4 are guaranteed to hold because the elementary tree cannot contain any open substitution nodes or predicted leaves before the integration site, and because the first future fringe $\underset{1}{P'}$ has to have the form $[\underset{1..n''}{A''}][\underset{1..o''}{B''}, L''] : nil$, i.e. the next leaf must be the lexical anchor, and the current fringe switches to the position after that leaf. Hence there is no possibility to introduce an open substitution node or predictive anchor, and

the current fringe again describes the path from the last lexical anchor (L'') to the next leaf, thus satisfying invariant 1. As in the other substitution operations, all indices from prefix and elementary tree are copied to the resulting tree, and invariant 5 is also maintained.

$$\frac{(C;\ -;\ 1..i;\ [\underset{1..m}{A}][]:nil;\ [])\ \bowtie\ (C';\ X';\ i+1;\ [][\underset{1..o'}{B'}]:S';\ [\underset{1..n''}{A''}][\underset{1..o''}{B''},L'']:nil,\underset{2..n'}{P'})}{(C';\ X';\ 1..i+1;\ \underset{2}{P'};\ \underset{3..n}{P'})}\ cat(S')=C$$

	past fringe	current fringe	future fringe
β	$\underset{1..n}{H}$	$[\underset{1..m}{A}][]:nil$	$[]$
ε	$[]$	$[][\underset{1..o'}{B'}]:S'$	$[\underset{1..n''}{A''}][\underset{1..o''}{B''},L'']:nil,\underset{2..n'}{P'}$
r	$[][\underset{1..o'}{B'}]:S'+\underset{1}{H},\underset{2..n}{H},[\underset{1..m}{A}][]:nil+[\underset{1..n''}{A''}][\underset{1..o''}{B''},L'']:nil$	$\underset{2}{P'}$	$\underset{3..n'}{P'}$
	$[][\underset{1..o'}{B'}]:S'+\underset{1}{H},\underset{2..n}{H},[\underset{1..m-1}{A},\underset{m}{A}+\underset{1}{A''},\underset{2..n''}{A''}][\underset{1..o''}{B''},L'']:nil$		

The proof for operation **SubstUpP** works analogously.

Adjoining Operations

Next, let's consider an adjunction operation. In adjunction, it is slightly more difficult to fit the fringes of the participating trees together in the correct order, because of the additional path from the root to the foot node. We do not want to go through every operation one by one because they are all very similar, and pick out **AdjDownP** as an example. We distinguish the cases where the foot is the rightmost child of the elementary tree from the case where it is the leftmost one.

AdjDownP (foot left)

When a tree with its foot to the left is adjoined into a prefix tree, the operation must happen at the right side of the adjunction node. To maintain the correct order of nodes in the past fringe, it is necessary to find the left side of the adjunction node inside the past fringe, and cut the node into its upper and lower halves, inserting the current fringe (i.e. the path from root to foot node) of the elementary tree. The insertion at the left side is symbolised by $\underset{\underset{j}{A}}{\oplus}$ to indicate that the insertion takes place at the left side of node $\underset{j}{A}$. The rest is reasonably straightforward: the path from the last leaf node up to the integration site $[\underset{1..j-1}{A},\underset{j}{A}]$ is joined with the first slice of the future fringe $\underset{1}{P'}\,\underset{k}{^{k}}$. Here, of course we only use the lower half of adjunction node $\underset{j}{A}$ to combine it with

the right hand side of the foot node, which is the first node of the $P'_{1\ k}{}^{k}$. The future fringe is then composed out of the remaining fringe slices from the elementary tree, whereby the right hand side of the foot node is melted with the upper half of the right side of the adjunction node A_j, followed by the rest of the prefix tree's current fringe $[\ A_{j+1..n}\ ][\ B_{1..o}\ ] : S$, and the future fringe of the prefix tree. This way, the nodes end up in depth-first search order on the flattened version of the resulting fringes, and invariant 2 is satisfied.

$$\frac{(C;\ X;\ 1..i;\ [\ A_{1..j-1}, A_j, A_{j+1..m}\ ][\ B_{1..o}\ ] : S;\ P_{1..n}) \bowtie (C';\ l;\ i+1;\ [][\ B'_{1..o'-1}{}^{k}_{k}, C'^{k}*\ ] : nil;\ P'_{1..n'}{}^{k}_{k})}{(C;\ X;\ 1..i;\ [\ A_{1..j-1}, A_j] + P'^{k}_{1\ k};\ P'_{2..n'-1}{}^{k}_{k}, P'^{k}_{n'\ k} + [\ A_{j+1..m}\ ][\ B_{1..o}\ ] : S, P_{1..n})}\ cat(A_j) = C'$$

foot left	past fringe	current fringe	future fringe
β	$H_{1..n}$	$[\ A_{1..j-1}, A_j, A_{j+1..m}\ ][\ B_{1..o}\ ] : S$	$P_{1..n}$
ε	$[]$	$[][\ B'_{1..o'-1}{}^{k}_{k}, C'^{k}*\ ] : nil$	$P'_{1..n'}{}^{k}_{k}$
r	$H_{1..n} \oplus_{j} [][\ B'_{1..o-1}{}^{k}_{k}, C'^{k}\ ] : nil$	$[\ A_{1..j-1}, A_j] + P'_{1\ k}{}^{k}$	$P'_{2..n'-1}{}^{k}_{k}, P'^{k}_{n'\ k} + [\ A_{j+1..m}\ ][\ B_{1..o}\ ] : S, P_{1..n}$

The bit of fringe which we insert into the history ($[][\ B'_{1..o'-1}{}^{k}_{k}, C'^{k}*\ ] : nil$) does not contain any substitution nodes or predicted *leaves*, which is important for invariants 3 and 4. The prediction nodes are not changed, thus satisfying invariant 5, and will eventually be turned into canonical nodes by the verification operation, even if they are not present on the current or future fringes any more. As argued in the **SubstDownP** operation, this operation also guarantees that the current fringe cannot be shifted over any open substitution sites or prediction leaves – it contains the path from the last lexical anchor (still the same as in the prefix tree) to the new next leaf, thus satisfying invariant 1.

AdjDownP (foot right)

If the foot is to the right, we will have to do an operation similar to the $\oplus_{j}$ operation on the past fringe, but to the future fringe. Since the adjunction takes place at a node that's open to its left, we'll have to find its right hand side in the future fringes, $P_{1..n}$, and insert the path from the foot node to the root node ($P'_{n'\ k}{}^{k}$, the last slice of the elementary tree future fringes) there. Again, we can see from the fringe order that the current fringe contains the correct path (invariant 1), and that the correct depth first search order is kept (invariant 2). Again, invariants 3, 4 and 5 are not violated.

$$\frac{(C;\ X;\ 1..i;\ [\underset{1..n}{A}][\underset{1..j-1}{B},\underset{j}{B},\underset{j+1..o}{B}]:S;\ \underset{1..n}{P})\ \bowtie\ (C';\ r;\ i+1;\ [][\underset{1..o'}{B'^k_k}]:S'^k;\ \underset{1..n'}{P'^k_k})}{(C;\ X;\ 1..i+1;\ [\underset{1..n}{A}][\underset{1..j-1}{B},\underset{j}{B}]+[][\underset{1..o'}{B'^k_k}]:nil;\ \underset{1..n'-2}{P'^k_k},\underset{n'-1}{P'}+[\underset{j+1..o}{B}]:S,\underset{1..n}{P}\oplus\underset{n'}{P'^k_k})}\ cat(\underset{j}{B})=C'$$

foot right	past fringe	current fringe	future fringe
β	$\underset{1..n}{H}$	$[\underset{1..n}{A},][\underset{1..j-1}{B},\underset{j}{B},\underset{j+1..o}{B}]:S$	$\underset{1..n}{P}$
ε	$[]$	$[][\underset{1..o'}{B'^k_k}]:S'^k$	$\underset{1..n'}{P'^k_k}$
r	$\underset{1..n}{H}$	$[\underset{1..n}{A}][\underset{1..j}{B}]+[][\underset{1..o'}{B'^k_k}]:S'^k$	$\underset{1..n'-2}{P'^k_k},\underset{n'-1}{P'^k_k}+[\underset{j+1..o}{B}]:S,\underset{n'}{P'^k_k}\oplus\underset{j}{\overset{A}{}}\ \underset{1..n}{P}$

The last operation to be examined and discussed in detail is **Verification**.

Verification The **Verification** operation may look fairly complicated – so how does it implement the invariants? The $\oplus_k$ operators guarantee to maintain the same order of nodes in the past fringe (since prediction and verification tree must have the exact same shape to the right of their respective spines, as tested by the $m(\varepsilon,\underset{all}{N^k_k})$ function). In the past fringe, the only thing that changes is that index k is removed from nodes. In the current and future fringes, correct order is again enforced by the $\oplus$ operator, but here, additional bits of fringes that are present in the verification tree but not the prediction tree, are inserted, any additional branches (and associated bits of fringes) that end in substitution nodes to the right of the spine are inserted at the correct matching position in the prediction tree. At the end of the verification operation, the correct node order in the fringes is therefore guaranteed (invariant 2), and all indices k disappear. Invariant 5 is guaranteed to be satisfied by the $m(\varepsilon,\underset{all}{N^k_k})$ function: all indices from the same original prediction tree are verified and thus removed at once, and no other indices are affected.

The prefix tree did not contain any open substitution nodes or leaf nodes before the word anchor w_i, but the leaf following w_i, which is visible on the prefix tree's fringe, is a prediction leaf. The elementary tree does not contain any prediction leaves itself, but may contain open substitution nodes before the lexical anchor w_{i+1}. Invariant 3, saying that there must not be any open substitution nodes before w_{i+1} is satisfied because we know that any substitution nodes to the left of the spine that are open in the verification tree were already filled while processing words $w_1..w_i$, otherwise they would appear on the current fringe, and the current fringe of the prefix tree would still be inside the prefix tree's future fringes. The open substitution nodes to the left of the spine in the elementary tree therefore get all filled during verification. Invariant 4 is

satisfied because the prediction tree leaf L_k^k is verified against the matching fringe in the verification tree, L''. The current fringe now contains the path from the last lexical anchor (L'') to the next leaf, which means that invariant 1 is satisfied.

$$\frac{(C;\ F;\ 1..i;\ [\underset{1..m}{A}][\underset{1..o}{B}, L_k^k] : nil;\ \underset{1..n}{P}) \bowtie (C';\ F';\ i+1;\ [][\underset{1..o'}{B'}] : S';\ \underset{1..i'-1}{P'}, [\underset{1..m''}{A''}][\underset{1..o''}{B''}, L''] : nil, \underset{i'+1..n'}{P'})}{(C;\ F;\ 1..i+1; \underset{1}{P} \oplus \underset{i'+1}{P'};\ \underset{1}{P} \oplus \underset{i'+2..n}{P'}, \underset{2..n}{P})} m(\varepsilon, \underset{all}{N_k^k})$$

foot right	past fringe	current fringe	future fringe
β	$\underset{1..n}{H}$	$[\underset{1..m}{A}][\underset{1..o}{B}, L_k^k] : nil$	$\underset{1..n}{P}$
ε	$-$	$[][\underset{1..o'}{B'}] : S'$	$\underset{1..i'-1}{P'}, [\underset{1..m''}{A''}][\underset{1..o''}{B''}, L''] : nil, \underset{i'+1..n'}{P'}$
r	$\underset{1..n}{H} \oplus_k [][\underset{1..o'}{B'}] : S', \oplus_k \underset{1..i'-1}{P'},$ $[\underset{1..m}{A}][\underset{1..o}{B}, L_k^k] : nil \oplus_k [A''][B'', L''] : nil$	$\underset{1}{P} \oplus \underset{i+1'}{P'}$	$\underset{1}{P} \oplus \underset{i'+2..n}{P'}, \underset{2..n}{P}$

End Condition

Finally, the end condition for valid PLTAG derivations says that a derivation is complete when there are no more open substitution sites or prediction nodes, and the root node is S. This becomes true when we reach a point where the future fringe is empty, and the last node on the current fringe has category S. Since there are no more prediction leaves on the fringe, all nodes that were annotated with an index must have been validated during verification procedures, because single left-over nodes would conflict with the $m(\varepsilon, \underset{all}{N_k^k})$ function. The operations that integrate a prediction tree always add at least one fringe item that is a prediction leaf, but their current fringes cannot move over it except during verification. Therefore, all prediction trees introduced with these rules must have been verified and no prediction nodes can possibly be left invalidated in the tree. Furthermore, the operations guarantee that it is never possible to have open substitution nodes to the left of the last anchor. Since there are none on the current fringe either, the condition that no open substitution nodes are left on the fringe is satisfied.

8.4 Optimisations for the Implementation

The introduction of the concept of a fringe is a first step to making the algorithm tractable, because it is not necessary to store the whole trees during the search phase of parsing, but just the current and future fringes. The search space is however much bigger in the PLTAG parser than in other parsers, because prediction trees are not lexicalised and can thus be used at any point in the sentence.

8.4.1 Restricted Use of Prediction Trees

An obvious optimisation is therefore to restrict the use of prediction trees. Otherwise, nothing prevents the parser from integrating prediction trees infinitely[3] without ever processing the next word.

In the implementation presented here, the prediction lexicon is restricted to prediction trees needed to parse the training set. A further restriction is to only allow them in constellations that were observed during training. The advantage of that further restriction is that we can pre-combine prediction trees, and can forbid the algorithm to integrate two prediction trees in a row, thus cutting down on the search space even further.

Only allowing one prediction tree to be integrated at a time however leads into the dilemma of either requiring an infinite set of (pre-combined) prediction trees (for left-recursive constructions as shown in Figure 8.26), or accepting slightly lower coverage of the parser. For example, the sentence "I love Peter's father's cousin's cat." shown in Figure 8.26, can easily be parsed non-incrementally, or with a parser that allows to integrate several prediction trees in a row, as any decent-sized lexicon will contain all elementary trees needed for this sentence. In order to parse this sentence incrementally with the restriction of using maximally one (pre-combined) prediction tree at a time, a structurally exactly identical case has to be contained in the training data to allow for the combination of six prediction trees, so that three nouns and three possessive markers can be predicted for correctly integrating *Peter*.

Because of recursive rules, embedding can in principle be infinitely deep, thus requiring infinitely many prediction trees to be applied in succession in order to parse a sentence. However, such structures do not usually occur in naturalistic examples of language usage. We found that in the whole of the Penn Treebank, at most five prediction trees had to be applied in a row for connectivity, and such a case only occurred once. In more than 95% of cases, only one prediction tree is needed.

From a psycholinguistic point of view, we can argue for limited human memory and could therefore constrain the size of prediction tree combinations to a specific number x. We would then avoid the infinity problem. Combinations of prediction trees that were not seen in the training data but contain less than x prediction trees could be generated automatically by combining all prediction trees (also with themselves), a maximum of x times. Note however that we would run into pretty bad combinatory

[3] In case of a parser using beam search, the limit would be that no new trees can be added without falling out of the beam width during pruning.

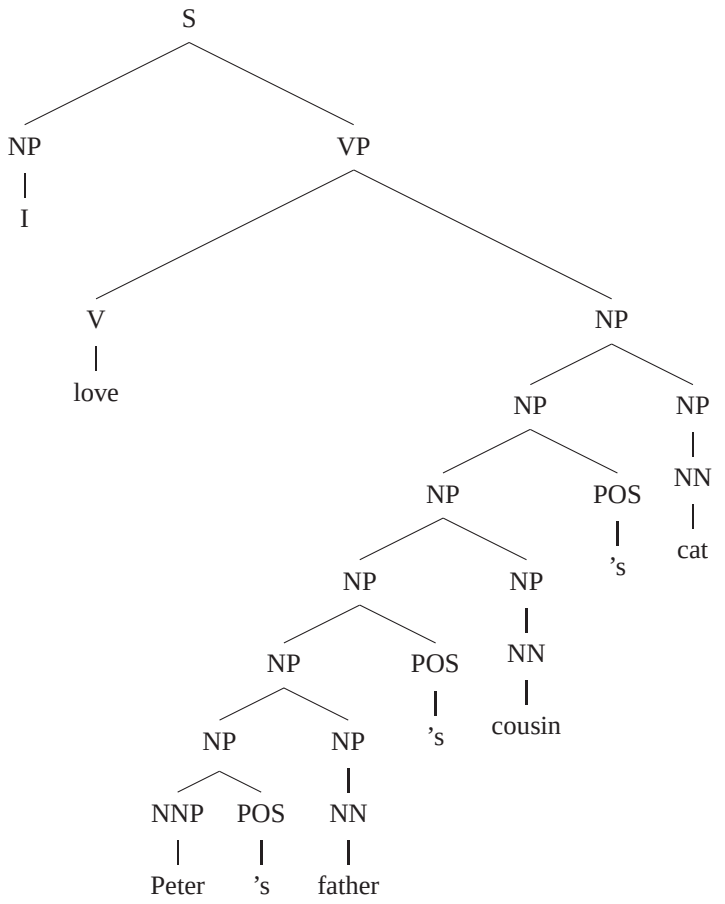


Figure 8.26: PLTAG parse tree for a sentence with left recursion.

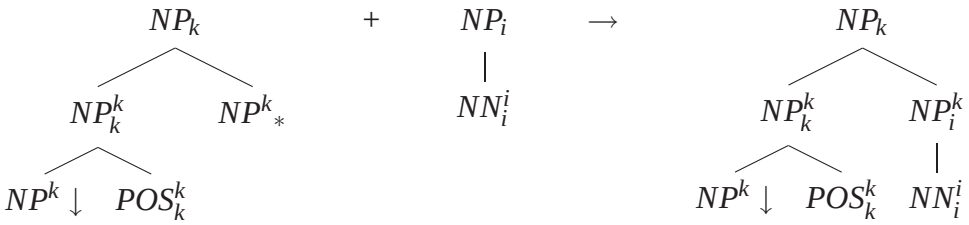


Figure 8.27: An Example of prediction tree indexing.

explosion of the prediction lexicon's size even if the maximal number of prediction trees that can be combined is restricted.

As explained in Section 7.2, all the nodes in a prediction tree are marked with indices. When prediction trees are combined, we have to make sure that the different parts of the trees are assigned indices that reflect which node was part of which tree originally. It is then very easy to keep track of which nodes need to be verified at once, and guarantee that the prediction tree will be fully verified. For an example of prediction tree pre-combination, see Figure 8.27.

An alternative would be to postulate a lazy prediction strategy. This lazy prediction strategy would only try to use prediction trees if the trees cannot be combined otherwise. However, this entails that we would have to let the parser do backtracking, since some analyses may need prediction even though there are other analyses (or analyses that only later turn out to be false) which do not necessitate prediction. In the case of a lazy strategy, predictions would only occur through connectivity and subcategorization.

8.4.2 Arguments vs. Modifiers

Arguments and modifiers are distinguished based on the PropBank annotation and more detailed PTB annotations, which contain markers such as "-CLR" for "closely related" etc. For cases where the PTB annotation differs from the PropBank annotation, the constituent is assigned modifier status, because lexicon size would increase significantly if all of these cases were encoded as arguments. A source for identifying arguments of nouns is NomBank. However, the relations annotated as arguments in NomBank tend to be semantic arguments, which are not required syntactically, and thus would lead to a much bigger lexicon and increased data sparseness problems. We therefore decided not to use the NomBank annotation.

8.4.3 Chart parsing

Chart parsers and the CKY algorithm were particularly successful because they calculate combinations of grammar rules only once and re-use structures for different analyses. In our incremental algorithm, a similar procedure can be implemented in order to avoid calculating integrations between trees with identical current fringes repeatedly.

The chart for the incremental algorithm consists of a table with the words $w_1 \ldots w_n$ on the one dimension, and a list of chart entries containing analyses that share the same current fringe (but can have different future fringes), as shown in Figure 8.28. For simplicity, the chart in Figure 8.28 only shows the current fringe but not the future fringes in the chart cell. Storing analyses with different future fringes in the same chart entry leads to a range of house-keeping issues, as the different future fringes must be associated with the correct subset of operations that created them, in order to retrieve correct analyses at the end of the search. For example, a high vs. low attachment decision can impact on the order of nodes in the future fringe, while the current fringe of the analyses is identical. So for some time, the two analyses will be treated the same,

but once the future fringes that contain the nodes in different order due to the previous high/low attachment have moved onto the current fringe, these analyses will end up in different chart entries and different operations may apply to them. At the end of the search phase, when the n-best trees are constructed, it is crucial to associate the analysis with the high attachment future fringe with the original high attachment operation in order to build valid trees. This situation is hinted at in Figure 8.28 in the third row in the column with the second *the*: Another analysis also points back to the same chart entry (as indicated by the red arrow), but was generated from a different analysis. The best analysis must thus always follow the path indicated by the black arrow, and the other one the path indicated by the red arrow. In order to correctly identify the best analysis, it is necessary to update the probabilities of all analyses within a chart entry at every step, i.e. not just one probability per chart entry, but as many probabilities as the number of analyses that the chart entry contains (keeping track of the probabilities of all analyses is necessary anyway in order to calculate prefix probabilities at each word).

In practice, this is implemented as follows: firstly, trees retrieved for the current word w_i are combined in all possible ways with prefix trees $\beta_{1..i-1}$, and the probability and construction history of each resulting analysis is updated: The probability of an analysis is stored at the last future fringe, in order to correctly associate each analysis with its maximum probability. Furthermore, a pointer to the elementary tree anchored in w_i and a pointer to the previous partial analysis $\beta_{1..i-1}$ are created, in order to be able to retrieve correct trees at the end. The analyses are then sorted based on their probabilities and pruned according to the beam width (this is the pruning step shown in line 9 of Algorithm 1). Next, the remaining analyses are added to the chart, thereby shifting the current fringe if necessary and combining any analyses with same current fringe to the same chart entry. For example, consider a simple determiner elementary tree (current fringes $[][DT, the] : nil$ and future fringe $[[the, DT][] : nil]$) being integrated with a chart entry with current fringe $[V, VP][NP] : DT\downarrow$, and two alternative future fringes, say $[DT\downarrow, NP, VP, S][] : nil$ and $[DT\downarrow, NP, VP, VP, S][] : nil$. As discussed in Section 8.3.3, the correct current fringe of the resulting tree is a combination of the first future fringe of the elementary tree and the first future fringe of the prefix tree. In our implementation of the parsing operations, the step of combining the future fringe of the prefix tree with the elementary tree's future fringe, $[the, DT][] : nil$ is performed after pruning (for efficiency reasons), when inserting the analyses into the chart. However, this means that two new current fringes are generated: $[the, DT, NP, VP, S][] : nil$ and

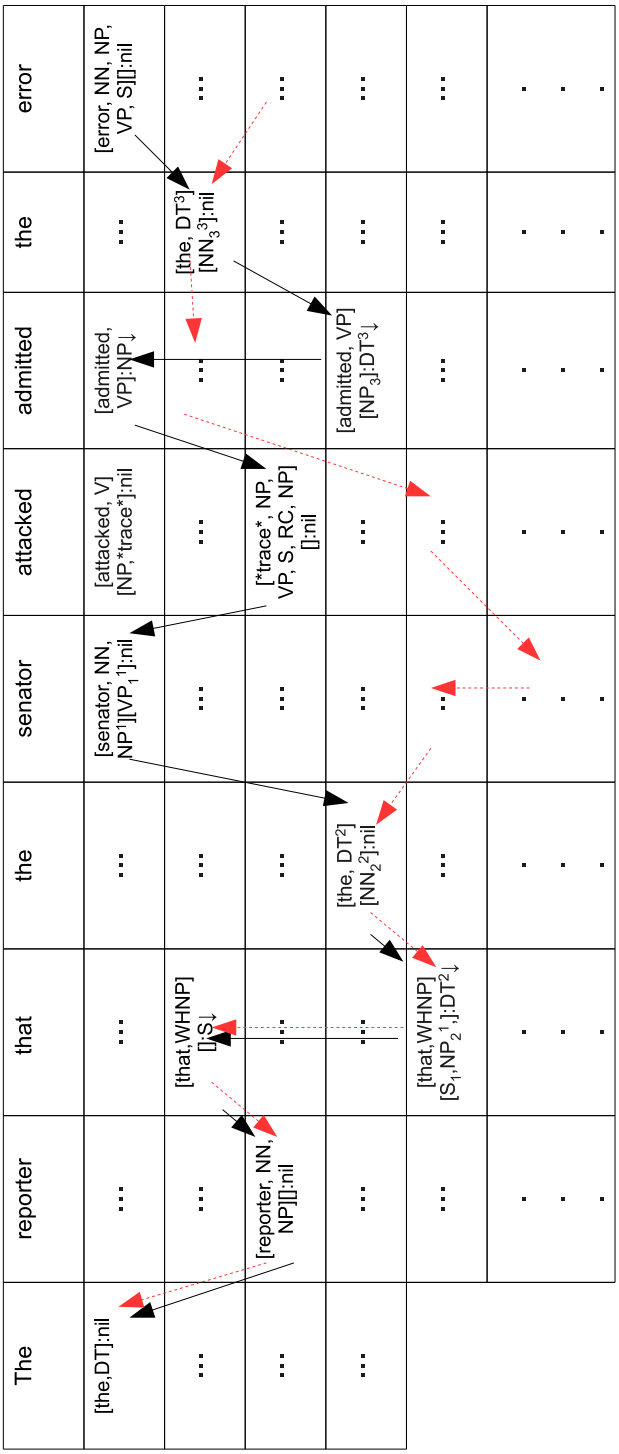


Figure 8.28: A chart for parsing the example sentence from Figure 8.15. The chart is shown after completion of the search phase. The black arrows show the sequence of operations and elementary trees needed to find the best parse for the sentence. The (red) dotted arrows show an example derivation path for a different analysis.

$[the, DT, NP, VP, VP, S][] : nil$. As these fringes are different, they need to be stored in different chart entries. We refer to this delayed joining of the elementary tree's future fringe with the prefix tree's future fringe as "expansion" (see line 11 in Algorithm 8.3.3). It is also possible that many of the analyses turn out to have the same current fringe after expansion, in which case they are stored in the same chart entry. If there are more expansions into different current fringes than identical current fringes, the number of chart entries can be larger than the beam. We therefore perform another pruning step (Algorithm 8.3.3 line 15) to again cut the number of analyses down to the beam width.

When search is completed, the best *n* trees are retrieved by following a chart entry's back-pointers to the partial derivations and elementary trees used to construct the analysis. However, since one chart entry can have different future fringes, only some of the back-pointers stored in a particular chart entry are compatible with a particular solution. Here is when the construction history (i.e. the pointers to the elementary trees which were integrated to construct the analysis) comes into play: only previous chart entries that are compatible with the elementary trees on the construction history are followed, see the black full vs. red pointed lines in Figure 8.28.

A cognitively more plausible parser would not have a separate search and tree construction phase, but construct full derived trees instead of just maintaining the fringes. The chart is thus more of an engineering step, which we hypothesise would not be necessary if the probability model was based on as much experience and world knowledge as humans have available, because good analyses could be chosen more accurately, and thus a much smaller beam would be sufficient, which in turn means that it would not be a problem to store the derived trees for all analyses.

8.4.4 Tagging

Currently, the parser operates on gold-standard tags. To maintain full incrementality, it would be necessary to use a fully incremental tagger without preview, or to let the parser retrieve the elementary trees based on the lexeme but not the POS tag. This would of course increase the ambiguity that the parser has to deal with. On the other hand, using a separate POS-Tagger without preview would yield worse POS-tagging results than state-of-the-art taggers, and can thus be expected to also have an impact on parsing quality.

8.4.5 Supertagging

Many TAG and CCG parsers use supertagging to cut down on the search space during parsing. Supertagging is also referred to as "almost parsing" and basically consists of choosing a small set of elementary trees to be integrated at each word, so that in the case of the supertagger choosing just one tree per word, the parsing process would only consist of joining the trees together. Supertagging thus increases the efficiency of the parsing task significantly.

Maintaining incrementality in parsing means that the supertagger should not be allowed to have any preview. However, lack of preview has detrimental effects on supertagging quality[4].

In the incremental PLTAG parser, the combinatory explosion due to a large number of trees to be integrated at each word is particularly grave, because prediction trees do not have any lexical anchor. The average ambiguity for integrating a prediction tree is thus the size of the prediction tree lexicon (i.e., 2800 for using the full prediction tree lexicon, or 700 for using only prediction trees that were seen more than 5 times in the training data), while the average ambiguity for canonical trees is 2.5 trees per type, and about 50 trees per token in parsing (since frequent words have lots of readings and very rare words are treated as unknown words for reasons of smoothing, and can therefore also have a large number of readings). Non-incremental parsers, which do not operate with any unlexicalized trees, thus incur a significantly lower level of ambiguity. In order to keep parsing times manageable, it turned out that it is necessary to select a subset of most promising prediction trees. So in fact the parser presented here does do supertagging, but only for prediction trees. Features for the estimation include the current fringe of the prefix tree f_β and the POS tag of the next word t_{w+1} to give a small look-ahead, which is important for supertagging performance. See the full probability model of the supertagger in Formula 8.1. The look-ahead of knowing the POS tag of the next word does not necessarily compromise incrementality, if we assume that the POS tag was determined without preview. It does however make the interpretation of prediction weaker: predictions caused by subcategorization frames are made more eagerly (after processing the head that subcategorizes them) than predictions necessitated for maintaining a fully connected structure, which are only made when the next word is perceived and processed enough to determine its POS tag. In general, it seems cognitively plausible to assume that only predictions are made that

[4] Personal communication with James Curran.

have proven useful in past experience, and are promising given the current context. It does however also mean that an additional process is assumed in human parsing: a fast heuristic component that selects promising predictions based on local information in addition to a deeper parsing process.

$$\sum_{\pi} P(\pi|\, f_\beta, t_{w+1}) = 1 \tag{8.1}$$

$$\text{where } P(\pi|\, f_\beta, t_{w+1}) = P(\pi|\, f_\pi, \lambda_\pi) P(f_\pi, \lambda_\pi|\, f_\beta, t_{w+1})$$

π = prediction tree	f = fringe t = POS tag
β = prefix tree	λ =category of spine leaf node

In order to alleviate data sparseness, a prediction tree is not directly conditioned on the fringe of the prefix tree and the next word's POS tag. Instead, the probability of the prediction tree π is estimated conditional on its fringe and the category of the category of the last node on its spine, λ; it's fringe and last spine node's category are in turn conditioned on the fringe of the prefix tree and POS tag of the next word.

Probabilities are estimated using maximum likelihood estimation.

$$\hat{P}(\pi|\, f_\pi, \lambda_\pi) = \frac{freq(\pi, f_\pi, \lambda_\pi)}{\sum_\pi freq(\pi, f_\pi, \lambda_\pi)}$$

$$\hat{P}(f_\pi, \lambda_\pi|\, f_\beta, t_{w+1}) = \frac{freq(f_\pi, \lambda_\pi, f_\beta, t_{w+1})}{\sum_{f_\pi} \sum_{\lambda_\pi} freq(f_\pi, \lambda_\pi, f_\beta, t_{w+1})}$$

In order to further reduce data sparseness, we estimate the probability on an alphabetically ordered set of unique categories of the current fringe instead of the exact current fringe of the prefix tree. For example, instead of conditioning on the fringe $[DT, NP, NP, VP, VP, VP, S][] : null$, we would condition on $\{DT, NP, S, VP\} : null$. The idea behind this probability model is that the order of nodes is less important than the identity of the nodes themselves as possible integration sites.

For smoothing, the Brants smoothing method, described in Section 8.5.1, is applied. We use one level of backoff for estimating $P(f_\pi, t_\pi|\, f_\beta, t_{w+1})$ on just the most probable integration point n_{f_β} instead of the fringe f_β

$$\max_{n_{f_\beta}} P(f_\pi, t_\pi | n_{f_\beta}, t_{w+1})$$

The reason for using the probability of the most likely integration point instead of the product of all possible integration points is that a fringe with more different categories

should not have a lower probability of a particular tree adjoining into it than a fringe with the same category on it but fewer different other categories.

The supertagger is integrated directly into the parser. After retrieving the prediction trees, the supertagger model is used to select the 10 best trees for each prefix fringe, which the parser then tries to combine with the partial analyses stored in the chart.

8.5 Probability Model

This section defines a probability model for the PLTAG parser. It specifies how PLTAG derivations are assigned probabilities. The probability estimate of a partial derivation at word w_i is the product of the probability estimate of the prefix tree covering words $w_1..w_{i-1}$ and the probability of integrating the elementary tree for word w_i using one of the PLTAG parsing operations.

Probabilities for elementary trees ε are estimated from sections 2 – 21 of the Penn TreeBank. To address data sparseness caused by unseen or rare events, the probability model has to be smoothed. We present two different smoothing models, and discuss backoff steps used in smoothing. Backoff steps are used in smoothing to get an approximate probability estimate for unseen events by using a similar (but more general) event.

For ease of reference, see the overview of the mathematical symbols that will be used for the definition of the probability model in this chapter in Table 8.3:

The details of the probability estimates for each parser operation differ slightly, and are outlined in the paragraphs below. The general pattern is the same for all operations: the probability of an elementary tree ε is conditioned on its integration site η_β in the prefix tree β, and is normalised with respect to all alternative elementary trees that could be integrated at this site. The conditional probability of an elementary tree given the integration site is thereby estimated as the independent probabilities of the tree template τ_ε conditioned on the integration site η_β, and the probability of the lexical anchor λ_ε conditioned both on the tree template τ_ε and, in order to capture bilexical probabilities, the lexical anchor of the head child of the integration site node λ_β.

$$P(\varepsilon|\eta_\beta) = P(\tau_\varepsilon|\eta_\beta) \times P(\lambda_\varepsilon|\tau_\varepsilon, \lambda_\beta)$$

Furthermore, the integration point node η_β is approximated by the tree template τ_η that originally introduced η_β, its lexical anchor λ_η, its category c_η, the position of η within τ_η, denoted as n_η, and the "trace mark" tm, which is a flag for whether the first and /

elementary trees	ε	(subsumes α, σ, π)
initial trees	σ	(subsumes π)
auxiliary trees	α	(subsumes π)
prediction trees	π	
prefix tree	β	
tree structures	τ	
integration point node[1]	η	(can be subst or adj node)
position of η within η's elementary tree	n	
category of integration point node	c	
a tree's head leaf	λ	(subsumes ζ, w, t)
category at λ (for prediction trees)	ζ	
word lexemes	w	
POS-tags	t	
trace mark	tm	

Table 8.3: Mathematical symbols used in the definition of the probability model.

or last node on the current fringe is a trace or null element.

$$P(\eta_\beta) = P(\tau_\eta, \lambda_\eta, c_\eta, n_\eta, tm)$$

The sentence processing model also includes a procedure for factoring in decay. This is however not part of the probability model itself, but of the linking theory, and will be discussed in Section 8.7.

Probabilities for Tree at First Word

The probability of integrating a particular elementary tree anchored in the first word using the **Start** operation is normalised with respect to the set of alternative elementary trees that can occur at the beginning of a sentence. In order to alleviate data sparseness issues, the probability of the elementary tree ε conditioned on the start of sentence symbol *SOS* is broken down into the probability of the tree template of the elementary tree τ_ε given the *SOS* symbol, and the probability of word w_ε being the anchor of tree template τ_ε. The probabilities for these two conditional events are estimated using maximum likelihood with frequency counts from the training section of the Penn TreeBank.

$$\sum_{\varepsilon} P(\varepsilon|SOS) = 1 \tag{8.2}$$

where SOS is start of sentence, $P(\varepsilon|SOS) = P(\tau_\varepsilon|SOS)P(w_\varepsilon|\tau_\varepsilon)$

$$\hat{P}(\tau_\varepsilon|SOS) = \frac{freq(\tau_\varepsilon,\ SOS)}{\sum_{\tau_\varepsilon} freq(\tau_\varepsilon,\ SOS)}$$

$$\hat{P}(w_\varepsilon|\tau_\varepsilon) = \frac{freq(w_\varepsilon, \tau_\varepsilon)}{\sum_{w_\varepsilon} freq(w_\varepsilon, \tau_\varepsilon)}$$

Probabilities for SubstDown Operations

The probability of substituting an initial tree into the open substitution site of the prefix tree is normalised with respect to all possible initial trees that could be substituted at this site, i.e. with respect to all initial trees with the same root category.

$$\sum_{\sigma} P_s(\sigma|\eta_\beta) = 1 \tag{8.3}$$

where $P_s(\sigma|\eta_\beta) = P_s(\tau_\sigma|\tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)P(\lambda_\sigma|\tau_\sigma, \lambda_\eta)$

$\lambda_\sigma = \zeta_\sigma$ if σ is prediction tree. $\lambda_\sigma = w_\sigma, t_\sigma$ if σ is non-prediction tree.
$\lambda_\eta = \zeta_\beta$ if β is prediction tree. $\lambda_\eta = w_\beta, t_\beta$ if β is non-prediction tree.

$$\hat{P}_s(\tau_\sigma|\tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm) = \frac{freq(\tau_\sigma, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)}{\sum_{\tau_\sigma} freq(\tau_\sigma, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)} \tag{8.4}$$

$$\hat{P}(\lambda_\sigma|\tau_\sigma, \lambda_\eta) = \frac{freq(\lambda_\sigma, \tau_\sigma, \lambda_\eta)}{\sum_{\lambda_\sigma} freq(\lambda_\sigma, \tau_\sigma, \lambda_\eta)} \tag{8.5}$$

We estimate the probabilities as outlined in equations 8.4 and 8.5 because treating the tree template conditioned on the prefix tree and the lexicalization of the tree template as independent events alleviates data sparseness issues. A very similar model for estimating TAG tree probabilities for parsing has also been successfully employed in Chiang's (2000) TIG parsers. The estimation of the lexical component shown in equation 8.5 is identical for all parser operations, therefore it won't be repeated in the following equations.

Probabilities for SubstUp Operations

The probability of an elementary tree ε (can be either initial or auxiliary tree) that is integrated into the prefix tree using a **SubstUp** operation is the probability of the prefix tree being substituted into the first substitution node of the elementary tree (i.e. normalization is based on the category of the first substitution node of the elementary tree).

$$\sum_{\varepsilon} P_s(\varepsilon|\eta_\beta) = 1 \tag{8.6}$$

$$\text{where } P(\varepsilon|\eta_\beta) = P_s(\tau_\varepsilon|\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm)P(\lambda_\varepsilon|\tau_\varepsilon,\lambda_\eta)$$

$\lambda_\varepsilon = \zeta_\varepsilon$ if ε is prediction tree. $\lambda_\varepsilon = w_\varepsilon, t_\varepsilon$ if ε is non-prediction tree.
$\lambda_\eta = \zeta_\beta$ if β is prediction tree. $\lambda_\eta = w_\beta, t_\beta$ if β is non-prediction tree.

$$\hat{P}_s(\tau_\varepsilon|\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm) = \frac{freq(\tau_\varepsilon,\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm)}{\sum_{\tau_\varepsilon} freq(\tau_\varepsilon,\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm)}$$

Probabilities for AdjDown Operations

The probability of an auxiliary tree being adjoined into the prefix tree is normalised based on the category of its root node and the possibility that no auxiliary tree of that type is adjoined to at all.

$$\sum_{\alpha} P_a(\alpha|\eta_\beta) + P_a(NONE|\eta_\beta) = 1 \tag{8.7}$$

$$\text{where } P(\alpha|\eta_\beta) = P_a(\tau_\alpha|\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm)P(\lambda_\alpha|\tau_\alpha,\lambda_\eta)$$

$$\text{and } P(NONE|\eta_\beta) = P_a(NONE|\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm)$$

$\lambda_\alpha = \zeta_\alpha$ if α is prediction tree. $\lambda_\alpha = w_\alpha, t_\alpha$ if α is non-prediction tree.
$\lambda_\eta = \zeta_\beta$ if β is prediction tree. $\lambda_\eta = w_\beta, t_\beta$ if β is non-prediction tree.

$$\hat{P}_a(NONE|\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm) = \frac{\sum_{all\eta_\beta nodes\ with\ no\ node\ adjoined} freq(\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm)}{\sum_{all\eta_\beta nodes} freq(\tau_\eta,\lambda_\eta,n_\eta,c_\eta,tm)}$$

Probabilities of AdjUp Operations

Similarly, the probability of an elementary tree ε being integrated using an AdjUp operation conditioned on the root category of the prefix tree is normalised with respect

to all elementary trees that contain a possible adjunction site with the same category on their current fringe, and the possibility that the prefix tree will not be adjoined into anything just now.

$$\sum_{\varepsilon} P_a(\varepsilon|\eta_\beta) + P_a(NONE|\eta_\beta) = 1 \tag{8.8}$$

$$\text{where } P(\varepsilon|\eta_\beta) = P_a(\tau_\varepsilon|\tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm) P(\lambda_\varepsilon|\tau_\varepsilon, \lambda_\eta)$$

$\lambda_\varepsilon = \zeta_\varepsilon$ if ε is prediction tree. $\lambda_\varepsilon = w_\varepsilon, t_\varepsilon$ if ε is non-prediction tree.

$\lambda_\eta = \zeta_\beta$ if β is prediction tree. $\lambda_\eta = w_\beta, t_\beta$ if β is non-prediction tree.

$$\hat{P}_a(\tau_\varepsilon|\tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm) = \frac{freq(\tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)}{\sum_{\tau_\varepsilon} freq(\tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)}$$

Probabilities for Verification

The probability of a canonical elementary tree being integrated using the verification operation is conditional on the prediction trees that match the structure of the canonical elementary tree, and is normalised with respect to only those other canonical trees that are also compatible with the predicted nodes.

In order to capture the head-argument bi-lexical dependencies that were not available at previous integrations involving the unlexicalized prediction tree, the lexical anchor of the verification tree is conditioned on the lexical head that the prediction tree was originally integrated with.

$$\sum_{\varepsilon} P_v(\varepsilon|\pi_\beta) = 1 \tag{8.9}$$

$$\textit{where } P(\varepsilon|\pi_\beta) = P_v(\tau_\varepsilon|\pi_\beta) P(\lambda_\varepsilon|\tau_\varepsilon, \lambda_{\pi_\eta})$$

$$\hat{P}_v(\tau_\varepsilon|\pi_\beta) = \frac{freq(\tau_\varepsilon, \pi_\beta)}{\sum_{\varepsilon \textit{ compat with } \pi_\beta} freq(\tau_\varepsilon|\pi_\beta)}$$

8.5.1 Smoothing and Backoff-levels

When estimating these probabilities, there are data sparseness issues. Many events are only seen very rarely, or not at all during training. However, we do not want to imply that an event that was not seen in the training data is impossible, and hence it should not be assigned probability zero. The standard approach to alleviating this

problem is to use some kind of smoothing. In smoothing, some probability mass from the observed events is re-distributed onto unobserved events. In order to do this in a way that differentiates between more or less likely events, it is common to use backoff for smoothing, which means that we estimate the probability of a given event using the probability of a similar event, usually by taking away some of the conditioning parameters (also referred to as deleted interpolation).

8.5.1.1 Backoff levels

The backoff-levels for PLTAG-parsing are adapted from Chiang (2000), see Table 8.4.

Backoff levels for $\hat{P}(\tau_\varepsilon \mid \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)$	
l1 $\hat{P}(\tau_\varepsilon \mid \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)$	no backoff
l2 $\hat{P}(\tau_\varepsilon \mid \tau_\eta, t_\eta, n_\eta, c_\eta, tm)$	removing the lexeme w_η from λ_η
l3 $\hat{P}(\tau_\varepsilon \mid \tau_\eta, n_\eta, c_\eta)$	removing POS tag, adj. position in fringe, trace marker
l4 $\hat{P}(\tau_\varepsilon \mid c_\eta)$	conditioning only on category of integration node
Backoff levels for $\hat{P}(\lambda_\varepsilon \mid \tau_\varepsilon, \lambda_\eta)$	
l1 $\hat{P}(\lambda_\varepsilon \mid \tau_\varepsilon, \lambda_\eta)$	no backoff
l2 $\hat{P}(\lambda_\varepsilon \mid \tau_\varepsilon, t_\eta)$	no bi-lexical probability, just integration POS tag
l3 $\hat{P}(\lambda_\varepsilon \mid \tau_\varepsilon)$	probability of lexeme estimated based on tree structure
l4 $\hat{P}(\lambda_\varepsilon \mid t_\varepsilon)$	probability of lexeme estimated from POS tag only

Table 8.4: Backoff-levels for the incremental LTAG parser. The probability of a tree is based on the product of two estimated probabilities, one for the tree structure, and one for the lexeme given the tree structure. Both of these probability estimates are smoothed separately, which means that they have separate back-off levels.

As has been shown before for other parsers (Bikel, 2004), bi-lexical probabilities (which are weighed in only in backoff level l1), have a very small influence on overall parsing accuracy, presumably due to data sparseness. For the PLTAG parser, they only account for .5% point accuracy gain.

8.5.1.2 Smoothing

For the parser presented here, we use standard smoothing methods to estimate unseen events. Our smoothing methods use interpolated backoff, where the values for different backoff stages (as outlined in Table 8.4) are interpolated – this means that for each estimate, even if we have an estimate for the most specific context, we use the probabilities

from the similar events as well and weigh all the probabilities using interpolation factors. Standard smoothing algorithms for interpolation between different backoff-levels include e.g. Witten-Bell Smoothing and Kneser-Ney Smoothing (or a variant called Modified Kneser-Ney Smoothing), and a smoothing algorithm originally developed for POS-tagging by Thorsten Brants (Brants, 2000), which achieved best performance when compared to the other smoothing algorithms in a German Parser presented in (Dubey, 2004). For the parser presented here, two different smoothing techniques were implemented and evaluated: Brants' smoothing and the smoothing technique used in (Chiang, 2000), which is similar (but does not optimize parameterization) to the smoothing technique used in (Collins, 1999; Bikel, 2004);

A big difference between the smoothing algorithm is that the Brants' algorithm uses fixed interpolation parameters that are independent of the context. This can be advantageous if the data is too sparse to effectively estimate the parameters in all necessary contexts. It therefore doesn't need a held-out set for estimating the interpolation parameters, making the implementation a bit less complex than e.g. Modified Kneser-Ney Smoothing, which achieved second best results for Dubey's German Parser. In Brants' Smoothing, the interpolation parameters (one is needed for each backoff level to weigh the influence of that backoff level) are estimated based on increasing the interpolation weights for a particular backoff level if that backoff level estimates the conditional probability best (in terms of maximising it) for the events observed during training. The algorithm for the estimation of the interpolation parameters $\lambda_1..\lambda_4$ is shown in Figure 8.29. The smoothed probabilities are estimated as:

$$p = \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 + \lambda_4 x_4$$

where $x_1..x_4$ are the different backoff levels shown in Table 8.4.

The smoothing model used in many other parsers, such as the Collins parser (Collins, 1999), Dan Bikel's reimplementation (Bikel, 2004) and David Chiang's TAG parser (Chiang, 2000), assigns a different smoothing term to each different context. The smoothing terms are calculated on-the-fly, using the following formula to calculate interpolation terms λ:

$$\lambda_i = (1 - \frac{d_{i-1}}{d_i})(\frac{1}{1 + \frac{5u_i}{d_i}})$$

where d_i is the frequency of the context at backoff level i as seen during training ($d_0 = 0$), and u_i is the number of unique outcomes for that context during training, i.e. how

Input: Map containing tuples $\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle$, $\langle \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle$, $\langle \tau_\varepsilon, \tau_\eta, t_\eta, n_\eta, c_\eta, tm \rangle$, $\langle \tau_\eta, t_\eta, n_\eta, c_\eta, tm \rangle$, $\langle \tau_\varepsilon, \tau_\eta, n_\eta, c_\eta \rangle$, $\langle \tau_\eta, n_\eta, c_\eta \rangle$, $\langle \tau_\varepsilon, c_\beta \rangle$, $\langle c_\beta \rangle$, and their frequencies

Output: Estimate for interpolation parameters $\hat{\lambda_1} - \hat{\lambda_4}$

$\hat{\lambda_1}, \hat{\lambda_2}, \hat{\lambda_3}, \hat{\lambda_4} = 0$;

foreach $\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle$ *with* $freq(\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle) > 0$ **do**

$$d1 = \begin{cases} \frac{freq(\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle) - 1}{freq(\langle \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle) - 1} & if\ freq(\langle \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle) > 1 \\ 0 & if\ freq(\langle \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle) = 1 \end{cases}$$

$$d2 = \begin{cases} \frac{freq(\langle \tau_\varepsilon, \tau_\eta, t_\eta, n_\eta, c_\eta, tm \rangle) - 1}{freq(\langle \tau_\eta, t_\eta, n_\eta, c_\eta, tm \rangle) - 1} & if\ freq(\langle \tau_\eta, t_\eta, n_\eta, c_\eta, tm \rangle) > 1 \\ 0 & if\ freq(\langle \tau_\eta, t_\eta, n_\eta, c_\eta, tm \rangle) = 1 \end{cases}$$

$$d3 = \begin{cases} \frac{freq(\langle \tau_\varepsilon, \tau_\eta, n_\eta, c_\eta \rangle) - 1}{freq(\langle \tau_\eta, n_\eta, c_\eta \rangle) - 1} & if\ freq(\langle \tau_\eta, n_\eta, c_\eta \rangle) > 1 \\ 0 & if\ freq(\langle \tau_\eta, n_\eta, c_\eta \rangle) = 1 \end{cases}$$

$$d4 = \begin{cases} \frac{freq(\langle \tau_\varepsilon, c_\beta \rangle) - 1}{freq(\langle c_\beta \rangle) - 1} & if\ freq(\langle c_\beta \rangle) > 1 \\ 0 & if\ freq(\langle c_\beta \rangle) = 1 \end{cases}$$

  switch *max(d1, d2, d3, d4)* **do**

    case $d1$: $\lambda_1 += freq(\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle)$

    case $d2$: $\lambda_2 += freq(\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle)$

    case $d3$: $\lambda_3 += freq(\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle)$

    case $d4$: $\lambda_4 += freq(\langle \tau_\varepsilon, \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm \rangle)$

  end

end

$\hat{\lambda_1} = \frac{\lambda_1}{\lambda_1+\lambda_2+\lambda_3+\lambda_4}$

$\hat{\lambda_2} = \frac{\lambda_2}{\lambda_1+\lambda_2+\lambda_3+\lambda_4}$

$\hat{\lambda_3} = \frac{\lambda_3}{\lambda_1+\lambda_2+\lambda_3+\lambda_4}$

$\hat{\lambda_4} = \frac{\lambda_4}{\lambda_1+\lambda_2+\lambda_3+\lambda_4}$

Figure 8.29: Brants' Smoothing for estimating the probability for tree structures $\hat{P}(\tau_\varepsilon | \tau_\eta, \lambda_\eta, n_\eta, c_\eta, tm)$. The smoothing works analogously for estimating $\hat{P}(\lambda_\varepsilon | \tau_\varepsilon, \lambda_\eta)$.

many different cases were conditioned on this context. Linear interpolation is done via a recursive term:

$$p = \lambda_1 l1 + (1\lambda_1)(\lambda_2 l2 + (1-\lambda_2)(\lambda_3 l3 + (1-\lambda_3)(\lambda_4 l4 + (1-\lambda_4)10^{-19}))$$

A comparative evaluation of the two smoothing methods is provided in Section 8.6.

8.6 Parser Evaluation

The parser was trained on Sections 2-21 and evaluated on Section 23 of the Penn Tree-Bank (only sentences of length 40 or less were used for training and evaluation). We report Parseval labelled bracketing scores with respect to the PLTAG converted trees. This means that our results are not directly comparable with labelled bracketing scores obtained on the unconverted Treebank. We have also experimented with converting our TAG structures to flattened tree structures where a category never has the same category as a child. The flattened structures are a bit flatter than original Penn Tree-bank structures. When evaluating on the flat structures, F-scores decrease by about two points. This lower F-score is due to the fact that there are fewer brackets in total. To compare to other parsers, we also converted the output of the Charniak (2000) parser into this flatter format and found F-scores to also go down by 2% points with respect to non-flattened tree structures.

Coverage Out of the 2294 sentences of section 23 of length 40 or less there were 33 sentences (about 1.4%) that could not be successfully converted into PLTAG format. We therefore exclude these sentences from our analysis. Furthermore, there were 140 sentences for which no parse could be found within reasonable time/memory usage (10 min, 1.8 GB RAM), yielding a coverage of 93.80%. The reason for the failure to cover a sentence can be that all valid parse have fallen out of the beam, that a necessary prediction tree has not been selected by the supertagger, or that no grammatical parse can be derived given the PLTAG lexicon learnt during training.

Parsing Accuracy Table 8.5 gives the parsing results for the variants of the PLTAG model that we evaluated. The baseline model selects the most frequent parse for a given sentence: it adds up the frequencies of all the canonical and prediction trees for each parse and prunes low-frequency ones; the complete tree with highest overall frequency is returned as the best one. This baseline model achieves an F-score of 48.06 which serves to illustrate the difficulty of the task.

The full PLTAG probability model achieved an F-score of 64.40 with Witten-Bell smoothing, thus clearly outperforming the frequency baseline. A significant gain is achieved by replacing Witten-Bell with Brants smoothing, resulting in an F-score of 72.05 (all other parameters were held constant, beam width for both smoothing methods was set to 10^{-11}).

Model	Prec	Recall	F-score	Cov
Baseline	44.39	52.38	48.06	85.1
WB smoothing	62.63	66.28	64.40	93.8
Brants smoothing	72.73	71.38	72.05	93.8
Oracle	72.86	74.26	73.55	93.8
beam size=20	70.06	72.02	71.02	81.3

Table 8.5: Parsing results for the PLTAG parser; Baseline: frequency baseline for parse operation selection; WB: Witten-Bell; Oracle: correct prediction tree given

We also investigated the influence of bi-lexical probabilities and found that these only have a small effect on overall parsing performance, which only decreased by 0.5% when bi-lexical probabilities were removed from the model. Presumably this is due to data sparseness, as well as the fact that PLTAG lexicon entries are relatively large and already encode argument positions. The small effect of bilexical probabilities is consistent with previous results (Bikel, 2004). Interestingly, an extremely small beam size of only 20 chart entries with maximally 20 analyses per chart entry yields very similar results in terms of accuracy. However, only 81% of sentences can be assigned an analysis using this small beam width, see Table 8.5.

Parsing speed for our parser increases superlinearly in the number of words, see Table 8.30, which shows parsing times for a constant beam width of maximally 400 analyses. Even though the parser only tries to combine a limited, constant number of fringes (bounded by the beam width) against a limited number of elementary trees at each word (bounded by how many elementary trees the lexicon contains for a specific word, or, respectively, by the number of trees the supertagger offers to the parser), parsing times are not linear in the number of words. This can be explained by the fact that the fringes tend to get longer, thus providing more possible adjunction sites at the end of longer sentences. Note that parsing speed would be much higher if supertagging was introduced for canonical trees, thus strongly reducing the number of elementary trees that can be combined with prefix trees at each word.

8.6.1 Prediction Trees and Supertagging

The size of the prediction tree lexicon, and the set of prediction trees selected by the supertagger, influences parsing performance.

Let’s first consider the coverage of the prediction lexicon: In section 23, about

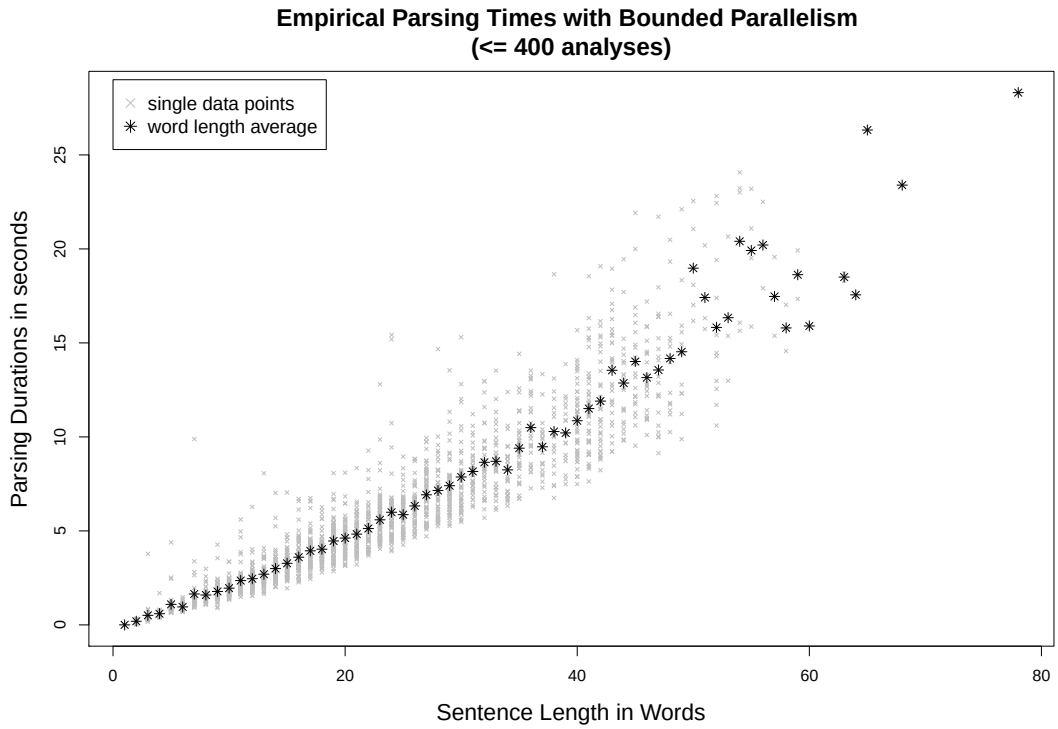


Figure 8.30: Empirical processing times as measured on a 2GHz, 1GB RAM machine.

4.5% of the sentences do not need any prediction trees to be parsed, and for 92.5% of the sentences, all the required prediction trees were seen in the training set with a frequency of more than five (our frequency cut-off). This means that we cannot parse the remaining 3.5% of sentences correctly even if all the required canonical trees have been seen. Furthermore, the supertagger might not select the correct prediction tree, and the parser would thus not be able to use the correct prediction tree even if it is contained in the lexicon.

But how well would we do if the prediction lexicon contained all necessary prediction trees, and if we always selected the correct prediction tree? To quantify the loss in F-score due to supertagging errors or missing prediction trees, we evaluated the parser using an oracle that always supplies the correct prediction tree. This increased the F-score to 73.55, see Table 8.5.

8.6.2 Comparison to other Parsers

Our results are not directly comparable to parsers that reproduce the Penn Treebank bracketing as our parser produces deeper tree structures informed by PropBank and Vadas and Curran's (2007) noun phrase annotation. We therefore compare to other TAG parsers only, but these also differ in which variant of the formalism they use

(LTAG, spinal LTAG, LTIG), resulting in F-scores that are not fully comparable.

Table 8.5 gives the F-scores of existing TAG parsers and compares them on the dimensions that are psycholinguistically relevant (incrementality, connectedness, prediction). The formalism that comes closest to ours in terms of psycholinguistic properties is Mazzei et al.'s (2007) DVTAG, for which however no implementation, probability model, or evaluation are available. All the other approaches achieve a higher F-score than our PLTAG parser, but at the cost of not being incremental (Chiang, 2000; Sarkar, 2001) or not building connected structures (Shen and Joshi, 2005). There also exist incremental fully-connected PCFG parsers which achieve better f-scores than our parser (84.4 – 87.4). The best-performing parser (Collins and Roark, 2004) uses a discriminative model, which is unsuitable for calculating prefix probabilities. Kato and Matsubara's (2009) parser is similar to Roark's (2001b) incremental top-down PCFG parser, which we used in earlier chapters to evaluate Surprisal. These two parsers seem to come closest to our parser in that they are incremental, construct fully connected structures on a word-by-word basis, and use a generative model which can be used for calculating Surprisal. They do however not satisfy the other requirements posed by our sentence processing theory, in particular modelling of prediction and verification processes.

Model	incr	con	pred	F
Mazzei et al. (2007)	+	+	+	N/A
This work	**+**	**+**	**+**	**72.1**
Sarkar (2001)	–	–	–	79.8
Chiang (2000)	–	–	–	86.7
Shen and Joshi (2005)	+	–	–	87.4*

Table 8.6: Comparison of this work with other TAG parsers; incr: incrementality; con: connectedness; pred: prediction; F: F-score; *: evaluated on dependencies.

8.6.3 Discussion

Differences in performance with other TAG parsers are likely due to the incrementality restriction (incremental parsers generally have slightly lower performance), not doing any supertagging for canonical trees, a large lexicon, and a sparse probability model. The sparse probability model is due to the large lexicon and the larger range of operators (`Up`/`Down` versions of substitution and adjunction, and verification). A further

effect of the prediction and verification mechanism is that many lexical dependencies are lost when prediction trees are integrated. Because prediction trees are not lexicalized, the statistical model cannot condition on the lexeme, but only on the prediction anchor (i.e., an internal category or POS tag). At verification, we are not currently taking into account all dependencies between the current word and the lexemes that had been integrated into the prediction tree. An improvement in parsing performance is likely to result from addressing this shortcoming. A discriminative model could possibly also yield improved f-scorses, but psycholinguistic measures like Surprisal cannot be calculated based on a discriminative model, as it conditions on the words, while Surprisal expresses in how far a word is unexpected.

This parser was however not designed for performance, but as the basis for a psycholinguistic model. In fact, many would have said before that a parser like the one presented here, which uses unlexicalized trees to make predictions in order to spell out the structure needed to connect all words would not be tractable at all. The 93.8% coverage and 72.1 point f-score mean that the parser is well suited for evaluating the sentence processing theory it was designed for on broad coverage text.

8.7 Formalisation of the Linking Theory

The desiderata for a linking theory: incrementality, connectedness, prediction with verification and parallel processing, were outlined in Section 6.2. Here, we formalise the linking theory with respect to the implementation of the PLTAG parser.

During processing, the elementary tree of each new word ε_{w_i} is integrated with all previous structures ($\beta_{w_1 \ldots w_{i-1}}$), 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). The trees (different prefix analyses and alternative elementary trees) have probabilities that express how good an analysis they are – from a psycholinguistic viewpoint these probabilities can be thought of as the analyses' prominence in the mind.

Each of the nodes of these predicted trees π has a time-stamp t that encodes when it was first predicted, or last activated (i.e., accessed). Based on the time stamp, a tree's nodes' decay d at verification time is calculated, under the assumption that recently-accessed structures are easier to integrate than more decayed ones.

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 (8.10) below). This surprisal component corresponds to the difficulty incurred through the parsing process. In addition to this, D has a second component, the cost of verifying earlier predictions, which is subject to a decay d (see the second line of Equation (8.10)). While the verification of prediction trees happens as part of the parsing process, difficulty is associated with retrieving previous predictions from memory, which is assumed to be a separate process from the parsing procedure. The overall processing difficulty D at word w_i is therefore:

$$\begin{aligned} D_{w_i} &= -\log \sum_{\beta_{w_1 \ldots w_i}} P(\beta_{w_1 \ldots w_i}) + \log \sum_{\beta_{w_1 \ldots w_{i-1}}} P(\beta_{w_1 \ldots w_{i-1}}) \\ &\quad -\log \sum_{\pi} P(\pi)^{(1-d^{t_\pi})} \end{aligned} \tag{8.10}$$

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, thanks to strict incrementality and a generative model.

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. Furthermore, verification cost depends on the probability of the prediction tree, while integration cost is independent on the probability of the structure of the head. Larger structures with more dependents tend to be less probable though, such that high verification cost for complex argument structures can still correlate with high integration cost due to several arguments needing integration.

8.7.1 Parameters in Theory and Implementation

The essence of the sentence processing theory proposed in this thesis is that humans predict upcoming structure, and that verifying the predicted structure causes processing difficulty (this is theoretically motivated by memory retrieval costs for remembering the prediction and integrating past information with the new structures). An important contribution of this work is to model the processes of prediction and verification explicitly. These processes of prediction and verification can in principle be modelled on top of a range of parsers and grammar formalisms, PLTAG as suggested in this

thesis is one implementation that realises the assumptions. Beside basic implementational choices like the grammar formalism, there is a number of further factors and parameters that modulate predictions of the theory:

- the decay factor
  The decay factor determines the rate at which predictions are "forgotten" and subsequently incur higher cost at being retrieved and matched against a verification tree. This also means that the decay factor influences the weighing between verification cost and the Surprisal component. With a low decay factor, verification costs could thus occupy a much larger value range than Surprisal values and hence be the main influencing factor in processing difficulty predictions. Similarly, with a decay rate that's very close to 1, verification costs would always be very small, and hence hardly change the predictions made by the Surprisal part of the equation.

- ticking of the clock / how to calculate the distance between head and argument
  As a simple assumption, we suggest to count distance in words, but that is probably not the best measure. An alternative are e.g. discourse referents[5]. Deciding on how to count distance has a similarly big effect on predictions as the decay factor. Another question is how time stamps should change. Should they be updated when something is integrated at them, thus accounting for re-activation effects? This would correspond to a reactivation of the head into which a new structure is integrated.

- beamwidth of the parser
  Parsing beam width influences predictions not only in how likely the parser is to achieve an accurate parse in the end. It also influences Surprisal values in that using a beam during parsing means that the prefix probabilities can only be approximated, but not calculated exactly because not all analyses are constructed and summed over when determining prefix probabilities of a structure. Secondly, the beam width also affects verification cost. The fewer analyses there are, the fewer verifications need to be executed.

[5]Discourse referents have been shown to be an imperfect measure. Alexopoulou and Keller (2007) show that two types of extraction from *wh*-phrases can differ in processing complexity, even though they involve the same number of intervening discourse referents. Based on this result, they argue that the number of intervening syntactic heads (rather than discourse referents) is the crucial factor for determining integration cost.

- shape of trees in lexicon

  Another variable is the shape of the trees. Here, we assume tree shape as motivated by linguistic theories, using standard TAG trees (with a few exceptions). But linguistic theories also differ with respect to how structures are analysed, what is regarded as a head etc. For example, if we regarded the determiner as the head of a noun phrase, a noun phrase would not need to be predicted when encountering a determiner, and no verification would happen when encountering the noun. Another aspect is the domain of locality, i.e. which lexemes are stored in the same lexicon entry. In its current implementation, only particle verbs like "show up" and "either..or"-like constructions are encoded in the same tree.

- training data / probability model

  The amount and type (i.e. what type of text is used for training, whether it is from the same domain as the target data that the model is to be evaluated on) of training data has a direct effect on parsing accuracy. Low parsing accuracy means that Surprisal estimates are imprecise because analyses are associated with incorrect probabilities, hence also leading to incorrect estimations of changes in probability mass. In addition, verification costs are adversely affected if there are many wrong analyses (in particular, analyses that seem very far-fetched from a human perspective) which contribute verification events and lead to unjustifiably high integration cost, or fail to assign high integration cost where it occurs because the correct analysis (or analysis preferred by a human at that point) has fallen out of the beam.

8.7.2 Implementation of Surprisal Component

Surprisal is calculated as the difference between the prefix probability at the current word w_n and the prefix probability at the previous word w_{n-1}. The prefix probability is by definition the sum of the probability of all trees covering words $w_1..w_n$. Because the parser is not doing a full search but using a beam and a supertagger for prediction trees for efficiency reasons, prefix probabilities calculated by adding up the probabilities of all analyses that fall inside the beam is only an approximation. To lessen the effect of beam search as much as possible, we calculate prefix probabilities at each word before pruning (i.e. prefix probabilities are calculated between line 8 and line 9 in Section 8.3, Algorithm 1).

8.7.3 Implementation of Verification Cost Component

The intuition when formulating verification cost is to capture the cognitive effort observed e.g. in English relative clause processing, long distance dependencies and centre embedding. These locality effects suggest that people incur difficulty when integrating new material under certain conditions. The theory presented here explains these difficulties as a result of matching new material against previously predicted structures. However, the processing theory assumes a language processor at human performance level, i.e. which is much better at language processing than any current parser, thanks to more exposure to data, semantic and world knowledge etc, which help to make more accurate predictions and analyses. If the implemented model had these additional resources, we hypothesise that it could accurately parse using a much smaller beam.

Beam size plays an important role for the estimation of verification costs – in the current parser setting, about 400 different current fringes are maintained at the same time, many of them containing multiple analyses. If we add up the verification cost for each verified tree, verification costs will sometimes be extremely high. When inspecting these cases, it turns out that most of the verifications in fact concern the same original prediction tree, which mostly coincides with what happens at the correct analysis. In addition, there are a number of "freak" analyses that seem very far-fetched from a human perspective and mostly have low probability. Given that they contribute a disproportionally large amount of verification cost to the total verification cost when verification costs from all analyses are summed up, it seems that a better estimate of the actually incurred verification cost would be to either weigh the verification costs by the probability of the analysis in which they occur, or to only count the verification cost incurred in the most likely analysis (which requires verification). Both of these approaches are also more compatible with the assumption of parallel processing than summing up the verification cost of all analyses. After implementing both of these approaches, it turned out that processing difficulty predictions were very similar (correlation of $r > 0.97$), and it was hence undecidable which version is better (it did not make any difference in either the case studies or the broad-coverage evaluation reported in Chapter 9). We (somewhat arbitrarily) decide on using the difficulty estimate based on the most likely verification.

8.7.4 Discussion

The linking theory contains two mechanisms, the surprisal component which quantifies difficulty incurred through unexpected events and the verification component which captures memory retrieval effects when matching newly encountered structure against predicted structure. Surprisal thereby directly falls out of the calculations necessary for the parser's probability model, while the decay effects in verification cost are not part of the probability model. In future work, it would be desirable to integrate these two theoretical components more closely: verification cost should affect the choice of which analyses the parser follows up on (i.e. what remains in the beam), while memory retrieval processes should in turn affect the parsing process, for example via incrementally updating the parser's probability model (see Section 10.2.3).

8.8 Conclusions

This chapter started out by describing the conversion of the Penn TreeBank into PLTAG format and the automatic induction of the canonical PLTAG lexicon and the prediction lexicon. Next, an incremental parsing algorithm for PLTAG, which incrementally derives fully connected partial structures on a word-by-word basis was presented. The parsing algorithm has been proven to only produce valid PLTAG derivations. In order to make the parsing process fast enough for broad-coverage parsing, a number of steps had to be taken to optimise over the implementation of the straight parsing algorithm. These optimisations include restricting the use of prediction trees, pre-combining them into larger prediction trees and introducing super-tagging for prediction trees in order to select a small number of most promising prediction trees. Furthermore, a generative probability model is proposed, which will enable us to easily calculate Surprisal on a word-by-word basis. Finally, we evaluate the parser on the Penn Treebank. It achieves a coverage of 93.8% and f-score of 72.1%, making it suitable for broad-coverage evaluation of the sentence processing theory proposed in Chapter 6.

The final section of this chapter presented and discussed the formalisation and implementation of the linking theory. The final piece of research in this thesis, the evaluation of the sentence processing theory based on the incremental, fully connected predictive PLTAG parser will be reported in Chapter 9.

Chapter 9

Evaluation

This chapter describes the evaluation results for the proposed sentence processing theory using the incremental PLTAG parser trained on the Penn Treebank. We first discuss the parser's predictions for a number of established psycholinguistic results and show that the theory manages to model a wide range of effects, such as locality effects in relative clauses as well as prediction effects.

The second part of this chapter presents evaluation results for difficulty predictions on naturally occurring broad coverage text, the Dundee Corpus. The evaluation method on the broad coverage text is again linear mixed effects models, as discussed in Chapter 3. We then compare the predictive power of our theory to the theories evaluated on this data in Chapter 5, DLT integration cost and Surprisal.

Parts of the material in this chapter have been published at CogSci 2009 (Demberg and Keller, 2009).

9.1 Evaluation on Psycholinguistic Case Studies

This section evaluates the proposed sentence processing theory on a series of established experimental processing results from the psycholinguistic literature, and compares the theory's capacity of accounting for the experimental results against other sentence processing theories. The modelling results reported in the following sections are based on a decay factor of $d = 0.9$ and a beam width of analyses within 10^{-11} probability of the best analysis in order to get an adequate Surprisal estimate. The number of time steps was set to the number of intervening words. The probabilities for the PLTAG grammar were derived from the Penn Treebank (cf. Chapter 8).

9.1.1 SRC / ORC asymmetry

One of the classic sentence processing results is the finding that subject relative clauses (SRCs) as in (1-a) are easier to process than object relative clauses (ORCs) as in (1-b). Refer to Section 4.1 for an overview on previous work on relative clauses. A recent study by Staub (2010) asked the question of where exactly processing difficulty occurs within relative clauses, after observing that theories like Surprisal make different predictions from theories like Dependency Locality Theory. While both theories predict that object relative clauses are more difficult to process than subject relative clauses, DLT would predict the difficulty to occur on the embedded verb phrase, while Surprisal would predict higher difficulty to occur on the NP in the relative clause. Staub (2010) found evidence for increased difficulty in both regions, and hypothesises that Surprisal-type theories and DLT-type theories predict different aspects of processing difficulty (just like suggested in this thesis). Therefore, this experiment is particularly relevant for the evaluation of our theory.

Data

We evaluated our theory on the materials used in Staub's (2010) study, experiment 1. The 24 sentence pairs are designed such that both conditions contain exactly the same words, but with different word order such that one is a subject relative clause and the other one is an object relative clause. These patterns are in fact the same as in the traditional "The reporter that *attacked the senator / the senator attacked* admitted the error" sentences, see (1) for an example.

(1) a. The bus driver *who followed the kids* wondered about the location of a hotel.
b. The bus driver *who the kids followed* wondered about the location of a hotel.

Method

We compare difficulty predictions from our theory to observed reading time data from Staub's (2010) relative clause study. One problem of comparing difficulty predictions directly with reading times is the missing link in our theory of how difficulty is exactly reflected on reading times: our model makes no claims about how difficulty is reflected in fixation behaviour and reading times, and is agnostic with respect to how processing

difficulty relates to different reading measures.

In other parts of this thesis, where multiple regression is applied to determine whether an explanatory variable is a significant predictor of reading times, the aspect of skipping behaviour was taken out of the model by removing all skipped words, for mathematical modelling reasons (see discussion in Section 3.2.2). Given that we are in this case study mainly interested in whether we can replicate a significant difference, and are not doing any regression modelling, taking both fixation durations and skipping into account will give a more intuitive picture of the processing difficulty that participants incurred during the experiment. Staub (2010) uses slightly different reading measures than were used in this thesis, i.e. *go-past times*. Go-past times are defined as the sum of the durations of all fixations from the first fixation on a word (only counted if the region to the right of the word has not yet been fixated) until the word is left to the right. In particular, fixations to the left of the word that happen after a regression out of the critical region are also counted. Go-past times are the latest measure reported in Staub's (2010) study, and therefore presumably capture difficulty effects more completely than an early measure[1]. In order to approximate a general notion of difficulty from these measures, go-past reading times were multiplied with fixation probability (determined from 1-skipping rate from Staub (2010)), thus obtaining average go-past times.

Results

Figure 9.1(a) shows average go-past times from Staub's (2010) study. Numbers in Figure 9.1(a) vs. 9.1(b) and 9.1(c) should not be directly compared quantitatively, but rather qualitatively. Our model makes qualitatively correct predictions for all empirical findings, see Figure 9.1(b): we predict no effect on relative pronoun and main verb, and indeed there was no effect on either of these regions in the empirical data either. The model also correctly predicts the larger difficulty on the embedded verb of the object relative clause.

For the embedded NP, the model predicts differences in processing difficulty between the conditions only on the first word of the NP, the determiner. This is because the onset of the NP is unexpected, but once the start of the NP has been processed, a noun is strongly expected. The longer observed reading times at the noun in the ORC

[1] In Staub's study, first pass times showed no effect on the determiner, an inverse effect on the noun region (first pass times were faster for the ORC noun than for the SRC noun) and a significant effect on the embedded verb, with longer reading times on the ORC embedded verb.

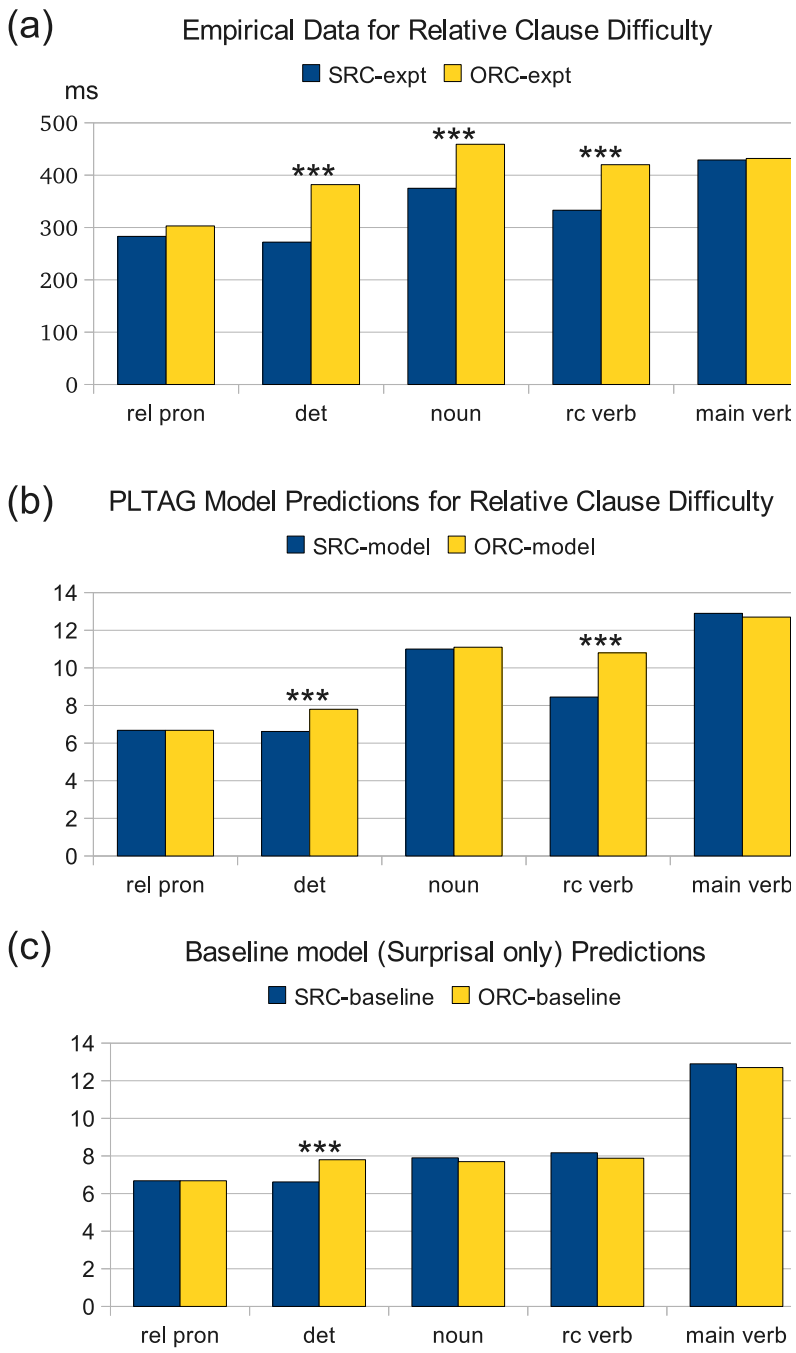


Figure 9.1: Staub (2010) experimental data vs. model predictions on the same materials for the different regions of the relative clause. Significance for $p < 0.001$ is marked as '***'. Subfigures (b) and (c) show predictions for the embedded verb and noun regions for the full PLTAG model and a Surprisal-only baseline model, respectively.

condition can be explained as a spill-over effect from the determiner, where the disambiguation between subject and object relative clause occurs. It is plausible to assume a spill-over effect at this point because the determiner was skipped frequently (48% of the time). As our model predicts processing difficulty but doesn't include any component for modelling how processing difficulty is reflected in reading times, we cannot model this spill-over effect.

Figure 9.1(c) shows the model predictions for the SRC and ORC sentences from Staub (2010) for a baseline model which does not take into account verification and only estimates processing difficulty in terms of Surprisal. The incorrect prediction of the Surprisal-only-baseline is consistent with Levy's (2008) observation that Surprisal is unable to predict the ORC/SRC asymmetry correctly. For the embedded NP region, predictions for the Surprisal baseline and a model including verification are qualitatively the same, with slightly larger difficulty predicted by the model including verification.

DLT integration cost makes the correct prediction of longer reading times on the ORC verb region, but does not predict any difference between conditions on the NP region. It therefore also explains the data less well than our PLTAG model.

Evaluation on Relative Clauses from the Dundee Corpus

In Chapter 4, we have shown that DLT integration cost can account for some of the reading time variance observed on naturally occurring relative clauses from the Dundee Corpus. Given that we have just shown that the PLTAG-based prediction theory proposed in this thesis can also account for the difference in processing difficulty in subject vs. object relative clauses, the question arises of whether it can also account for the processing difficulty on the embedded verb of relative clauses from the Dundee Corpus.

We ran a mixed effects model following procedures described in Sections 3.2 and 4.2, and included the predictions from the theory presented in this work, which we will refer to as PREDICTIONTHEORY, as one of the predictors in the regression model. Because PREDICTIONTHEORY is negatively correlated with WORDFREQUENCY, we used residualized PREDICTIONTHEORY values in the regression models, i.e. the part of PREDICTIONTHEORY which cannot be accounted for by word frequencies.

Residualized PREDICTIONTHEORY was a positive significant predictor of reading times on the embedded verb region of relative clauses both for log-transformed total reading times and log-transformed first pass reading times. For better interpretability,

Predictor	Coef	Sig
INTERCEPT	260.55	***
PREDICTIONTHEORY	15.16	**
WORDLENGTH	8.82	***
WORDFREQUENCY	-20.57	**
PREVIOUSWORDFREQUENCY	5.68	
LANDINGPOSITION	-73.20	***
LAUNCHDISTANCE	-4.28	***
WORDLENGTH:LANDINGPOSITION	-27.92	*

Table 9.1: Final regression model for total reading times on the embedded verb of relative clauses from the Dundee Corpus.

we report the result on raw total reading times, which is equivalent to log-transformed total reading times, in Table 9.1. The reported model includes a random intercept and random slope for WORDFREQUENCY under subject (all other random slopes lead to a decrease in model quality). Outliers were removed as usual (see Section 3.2.5).

But does PREDICTIONTHEORY work as well as INTEGRATIONCOST? Or does it work even better in predicting reading times? In order to answer these questions, we fitted a model that included all significant low-level predictors and both PREDICTIONTHEORY and INTEGRATIONCOST as explanatory variables in a log-transformed total reading time model, and compared this model against two models, each only containing the significant low-level predictors and one of the predictors. We found that removing INTEGRATIONCOST from the model including both INTEGRATIONCOST and PREDICTIONTHEORY did not significantly reduce model fit ($p = 0.16$; in fact the model including PREDICTIONTHEORY and not INTEGRATIONCOST was slightly better according to AIC and BIC). On the other hand, removing PREDICTIONTHEORY from the model including both INTEGRATIONCOST and PREDICTIONTHEORY did significantly reduce model fit ($p < 0.01$; AIC and BIC scores are lower (hence better) for the model including PREDICTIONTHEORY). This means that integration cost does not have a significant explanatory value above and beyond the predictions made by our theory. We conducted the same analysis on first pass reading times and found the same result.

Conclusion

The full version of our model as trained on the Penn Treebank correctly predicts the relative clause asymmetry pattern found in empirical studies (longer reading times in the verb region of the verb and noun regions of the object relative clause), as evidenced by running it on the 24 experimental items from Staub (2010). The Surprisal-only baseline of our model which does not associate the verification mechanism with any processing difficulty cannot account for the results on the embedded verb region.

Being able to replicate the relative clause data is particularly interesting, as previous models either predict difficulty on the NP region (like Surprisal) or on the verb region (like DLT), but not on both.

Furthermore, we showed the predictions by our theory also correctly account for reading times in naturally-occurring relative clauses from the Dundee Corpus, and that they explain the data better than DLT integration costs. This finding provides further support for our theory.

9.1.2 Either-or Predictions

The experiment reported in (Staub and Clifton, 2006) provides evidence for prediction in human sentence processing. The authors showed that following the word *either* readers predict the disjunction *or* and the complement that follows it; processing was facilitated compared to structures that include *or* without *either*. When *either* was present in items with sentence-level disjunction, it prevented people from initially misanalysing the sentence for NP-level disjunction. Such misinterpretations and following corrections only occurred in the sentence-level condition when *either* was not present.

Data

For our evaluation, we used the 48 example sentences from the Staub and Clifton (2006) study. As an example, consider the sentences in Example (2). Disjunction occurs at the noun phrase level in half the experimental items (like in Examples (2-a) and (2-b) and on sentence level in the other half of the materials (like in Examples (2-c) and (2-d)). Staub and Clifton (2006) found that the or-NP region (marked in recursive style in Examples (2)) is processed more quickly (i.e. first pass reading times and go-past times were significantly shorter) in (2-a) and (2-c) than in (2-b) and (2-d).

(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.
c. Either the student read a book *or his friend* wrote one.
d. The student read a book *or his friend* wrote one.

When running our model on these sentences, probabilities for the either conditions could not be estimated accurately because such uses of *Either . . . or* had not been seen in the training data (Sections 2–21 of the Penn TreeBank) and hence the model would back off a lot and give very low probability to seeing *or* following *either*. Occurrences of *either* in the Penn TreeBank are mostly "Either way . . ." constructions. We therefore added four sentences (one of each type) to the training data, making sure that the lexical items used were different from the ones in the test sentences. The sentences added to the training data were:

(3) a. The cat consumed (either) the food or the drink to get to the hut.
b. (Either) Albert lost a mobile or his colleague nicked it.

Adding at least a minimal amount of training data seems justified as one would not expect a human whose only language exposure is the Wall Street Journal, and who hence has not experienced the use of *either...or* constructions to exhibit typical reading time results on the experimental materials.

Results

Figure 9.2 graphs the predictions for the full model (Surprisal and verification components) for the *either . . . or* sentences from Staub and Clifton (2006). The graph shows the go-past reading times found experimentally for the or-NP region in the Staub study, compared to the sentence processing difficulty that our model predicts for this region. 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 effect was the same both in the experiment and the model, so the two conditions are merged together in Figure 9.2). 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 even lower difficulty predictions for the *either*-conditions.

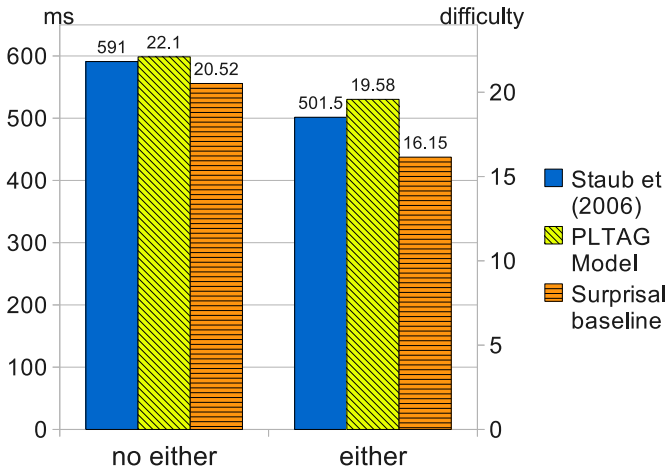


Figure 9.2: Average go-past time, average PLTAG prediction and average Surprisal baseline prediction for the *or-NP* region on the sentences used by Staub and Clifton (2006). The difference between the *no either* and *either* conditions is significant at $p < 0.01$ both for the model and in the experiment.

Discussion

Our model replicates the pattern found in the empirical study of either-or processing by Staub and Clifton (2006). The results demonstrate that our PLTAG model is not only able to replicate locality effects, as shown in the relative clause experiment, but also to capture prediction effects, which can be explained by Surprisal, but not by DLT.

As mentioned at the beginning of this section, the Staub and Clifton (2006) study not only found faster processing in the *or NP* region but also less misanalysis for sentence-level disjunction when *either* was present. Our implementation can also replicate this finding: In the sentence-level *either* case, the analysis predicting sentence coordination clearly was the most probable analysis (by 100 times) when processing *or*, which in turn also means that predicting additional structure to integrate the NP as the argument of an unseen verb still leads to a more probable analysis than if this additional structure had not been predicted and the NP had been integrated as an NP-level disjunction. In the case where *either* is not present, adjoining the structure for *or* at the NP level is by far the most probable analysis (because in the training data, NP-level disjunction has been seen more often than S-level disjunction). This analysis then allows for a direct integration of the NP into the *or* structure – having to predict a future verb makes the sentence-level analysis even less likely when the NP after *or* is processed.

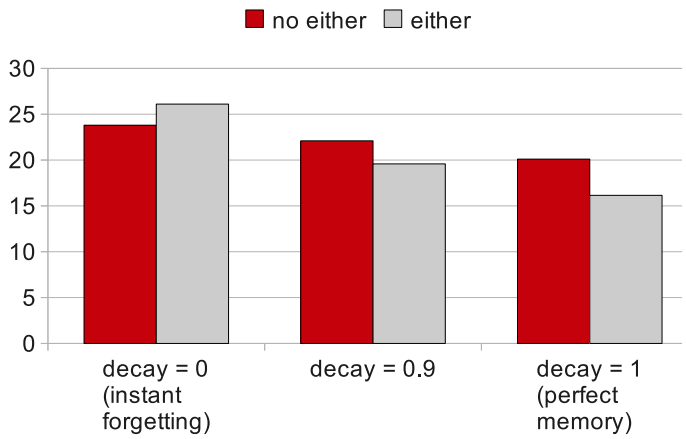


Figure 9.3: The influence of the decay parameter on modelling results for the *either. . . or* experiment.

Model predictions for this experiments depend also on the parametrisation of the model. The most important parameter is the decay factor, which (non-linearly) determines how the Surprisal part of the model and the verification cost are weighed, see Figure 9.3. If the decay factor is low, the model assumes that predictions are forgotten quickly, resulting in higher verification costs. Very high verification costs in turn lessen the relative effect of Surprisal on model prediction. If *either* was present, Surprisal is very low when *or* is encountered, in particular Surprisal is much lower than if *either* had not occurred previously. On the other hand, as *or* is integrated, some verification cost occurs in the *either* condition, but not otherwise. Whether the model makes the correct prediction thus comes down to weighing the difference in Surprisal between the conditions vs. the verification cost incurred at *or* in the *either* condition. For illustration see the model predictions for the *either..or* study for the chosen decay parameter 0.9 in comparison to the extreme decay parameters of 1 (which would correspond to perfect memory) and 0, which would in turn mean that the reader would not be able to remember any predictions as soon as the next word is processed in Figure 9.3.

The initial problems with this experiment, concerning the lack of exposure to *either.. or* constructions during training, highlight how much the modelling results also depend on the linguistic materials used in training. Modelling results will be more accurate and valid, if the model could be trained on a more varied and proto-typical language corpus than the Wall Street Journal.

9.1.3 Anti-locality Effects

Anti-locality effects have been shown for a number of languages, including German (Konieczny, 2000; Konieczny and Döring, 2003), Hindi (Vasishth and Lewis, 2006), and recently, also English (Jaeger et al., 2010). Anti-locality effects refer to the finding that reading times can be shorter at the head when intervening materials were inserted between the head and its arguments. Examples for anti-locality effects in German include experiments presented in (Konieczny, 2000) and (Konieczny and Döring, 2003). The original experiment reported in (Konieczny, 2000) found that the verb in verb-final constructions in German is read faster when more material (just one argument, vs. an additional PP vs. a longer, modified PP) occurred before the verb. This finding is contrary to the locality effect found in English centre embedding and the SRC/ORC asymmetry. A similar experiment was conducted by Konieczny and Döring (2003), who also controlled for length of the intervening material between conditions. An example of their materials is shown in (4).

(4) a. Die Einsicht, dass [NP-NOM der Freund] [NP-DAT dem Kunden] [NP-ACC das Auto aus Plastik] verkaufte...
the insight, that the friend the client the car from plastic sold, ...
The insight that the friend sold the client the plastic car . . .

b. Die Einsicht, dass [NP-NOM der Freund [NP-GEN des Kunden]] [NP-ACC das Auto aus Plastik] verkaufte, ...
the insight, that the friend of the client the car from plastic sold, ...
The insight that the friend of the client sold the plastic car ...

In materials following the example in (4), reading times on the verb *verkaufte* are shorter in (4-a) than in (4-b) even though the length of interfering elements is exactly identical. Surprisal can explain this finding as (4-a) restricts the possible identity of the head more strongly than (4-b). DLT would predict the opposite effect as one more argument is integrated in condition (4-a) than in condition (4-b). Similarly, the surprisal vs. verification components of PLTAG would make opposing predictions: while the surprisal component would predict less difficulty at the verb, the verification component would predict larger processing difficulty at the verb because it required a less probable prediction tree, which therefore at verification time will also be more difficult to verify than a more probable prediction tree. Whether the theory proposed here can correctly account for the observed effect is therefore inconclusive at this point - it

depends on the parametrisation of the model for German.

The English materials involve subject and object relative clauses with one, two or three optional PPs at the end, see Example (5-a) to (5-c). Reading times were measured on the region after the relative clause, i.e. the verb of the main sentence, *bought* in example (5).

(5) a. The player [that the coach met at 8 oclock] bought the house...
b. The player [that the coach met by the river at 8 oclock] bought the house...
c. The player [that the coach met near the gym by the river at 8 oclock] bought the house...

Jaeger et al. (2010) found that reading times on the critical region were faster the more PPs had been inserted at the end of the relative clause. This finding can in principle be explained by expectation-based theories such as Surprisal (because the expectation of the verb phrase grows stronger and stronger the more attachments are made, as fewer syntactic alternatives remain), but not by locality-based theories. The theory proposed here can potentially explain the difference in processing difficulty because its Surprisal component predicts faster reading times on the main verb, while the verification cost component predicts the same difficulty independent of the number of intervening PPs. This difference to DLT integration cost predictions stems from the fact that the main verb is not standardly predicted in PLTAG, and hence no verification costs occur, while DLT would predict increased integration cost for more PPs based on the larger number of intervening discourse referents between the main verb and its subject.

An open question is whether the theory could possibly explain the findings of Jaeger et al.'s (2010) second experiment, in which any PPs not occurring inside the relative clause are topicalised: in this case, the topicalisation of the PPs would trigger the prediction of a verb and hence cause verification cost later on in cases (5-a) and (5-b), but not in (5-c). The contrast between conditions (5-a) and (5-c) would be strengthened, but it is not clear whether the correct prediction could be obtained for case (5-b).

We ran the parser on all the experimental items provided in the appendix of Jaeger et al. (2010), but the difference in difficulty predictions on the main verb did not reach significance. Note also that the claim that Surprisal can explain the differences (Levy, 2008) is based on a single example, and that Surprisal values that are indeed very close: Levy reports Surprisal values of 13.87 for 1 PP, 13.54 for 2 PPs and 13.40 for 3 PPs, which are supposed to account for quite large differences in reading

times (510ms, 410ms and 394ms respectively). It is hence doubtful whether Surprisal can really account for the observed anti-locality effect.

9.1.4 Centre Embedding

Another well-known effect is *centre embedding* where an increasing number of nestings soon makes sentence structures impossible to process, see e.g. (Eady and Fodor, 1981; Chomsky, 1957). Consider the sentences from (Gibson, 1998) in example (6):

(6) a. The intern [who the nurse supervised] had bothered the administrator [who lost the medical reports].
b. The administrator [who the intern [who the nurse supervised] had bothered] lost the medical reports.

Sentence (6-b) has been shown to be considerably more difficult to process than (6-a) based on complexity ratings. In PLTAG, the more complex condition (6-b) would incur higher verification costs than the easy condition (6-a) because two verbs would have to be predicted simultaneously in (6-b), and high verification costs are incurred in particular at the second verb *had bothered*, as the distance to the initial prediction site is high. Furthermore, such double embedded structures are rare and therefore higher Surprisal costs are incurred than in the single embedding condition. When running the model on the example sentences in (6), the model replicated the preference for the easy condition by predicting lower processing difficulty than for the difficult condition.

9.1.5 Facilitating Ambiguity

Another effect that our theory can account for is *facilitating ambiguity*, as reported in (Traxler et al., 1998). The finding is that reading times can be faster under some circumstances in an ambiguous region than in an unambiguous region. Consider example (7): the reflexive pronoun (*herself / himself*) in (7-a) and (7-b) is unambiguous in that it can only refer to the daughter / colonel respectively. In sentence (7-c) however, *himself* is ambiguous as to whether it refers to the son or the colonel.

(7) a. The daughter$_i$ of the colonel$_j$ who shot herself$_{i/*j}$ on the balcony had been very depressed.
b. The daughter$_i$ of the colonel$_j$ who shot himself$_{*i/j}$ on the balcony had been very depressed.

c. The son$_i$ of the colonel$_j$ who shot himself$_{i/j}$ on the balcony had been very depressed.

Reading times were found to be faster on the *himself / herself* and immediately following region in the ambiguous case (7-c). This finding is difficult to account for under a locality or competition account. However, Levy (2008) explains how Surprisal can account for this effect: the attachment of the relative clause is ambiguous at *who* – it might attach to the *daughter/son* or the *colonel*. These two analyses are followed in parallel, but one of them is ruled out in cases (7-a) and (7-b), leading to higher Surprisal than sentence (7-c), where both analyses can still be maintained, resulting in lower Surprisal. The argumentation is exactly the same under the proposed theory, as the verification component makes no adverse predictions for this data.

Testing these sentences on the implemented theory is not possible as the parser does not make any checks to see whether *himself* would match *daughter* or not. It would hence not recognise the ungrammaticality of the low attachment in (7-a) and high attachment in (7-b).

9.1.6 Local Coherence Effects

It has been observed that processing difficulty can sometimes occur in sentences that are neither ambiguous nor particularly complex (Tabor et al., 2004). An example for this is the sentence in (8-a), which has the same syntactic complexity as (8-b).

(8) a. The coach smiled at the player tossed a frisbee by ...
b. The coach smiled at the player who was tossed a frisbee by ...
c. The coach smiled at the player thrown a frisbee by ...
d. The coach smiled at the player who was thrown a frisbee by ...

The important difference between the sentences is that in (8-a) the word sequence *the player tossed a frisbee* is a coherent string of words where *the player* would be the subject of a main verb *tossed*, while *the player thrown a frisbee* cannot be interpreted as such. While (8-a) can be expected to be the most difficult sentence among the sentences in (8), because reduced relative clauses are more difficult than non-reduced relative clauses, and ambiguous verb forms are more difficult than unambiguous verb forms, the observed difficulty effect was stronger than would be expected by adding the verb ambiguity and reduced relative clause effects. The common explanation for the

effect is that the locally coherent interpretation of *the player tossed a frisbee* interferes with the globally coherent analysis of the sentence, and has therefore been argued to provide evidence against a view of strictly incremental processing, as the locally coherent analysis should not be calculated in the first place, because *the player* already has a different function in the sentence, and cannot possibly be the subject of *tossed*, and *tossed* cannot be the main verb of the sentence, as there is already a main verb, *smiled*.

One explanation of the effect that would still be compatible with strictly incremental processing is the one suggested by Gibson (2006), who suggests that the observed effect might be due to a conflict between the top-down analysis which would require that *tossed* be analysed as the first word of a reduced relative clause (verb past participle), and a bottom-up analysis which would assign the most likely POS-tag to *tossed* using a unigram model, resulting in predicting a verb in simple past tense. The interaction effects of the ambiguity of tossed and the reduced relative would thus stem from the incompatibility of the most probable POS-tag for *tossed* and the globally coherent analysis.

From a PLTAG point of view, POS-tagging is not a step of just choosing the POS-tag. Instead, elementary trees are retrieved for each word. The implemented parser currently uses gold-standard POS-tags to reduce the ambiguity at parsing, therefore we cannot test the phenomenon at this point. However, our implementation could possibly account for the effect if super-tagging was introduced for the retrieval of elementary trees from the canonical lexicon. The explanation would then be that the super-tagger would fail in the difficult cases to provide the parser with the necessary reduced relative clause tree in the first place, and difficulty would ensue from the parser attempting to integrate the unsuitable tree(s) and on failing to succeed having to “ask” the super-tagger for more alternative elementary trees. The postulation of a supertagger for canonical trees would be a small theoretical step given that a supertagger for prediction trees has already been introduced to the parser model. Theoretically it would however mean to assume a separate heuristic mechanism in addition to the parsing process which quickly selects most promising syntactic structures based on local information only. Such an approach would be less parsimonious than an architecture that explains the same effect without local heuristics (e.g. the bottom-up parsing model of Morgan et al. (2010)), but it is conceivable that the human language processing system also uses fast local heuristics in addition to a more involved integration process.

9.1.7 Digging-in Effects

Digging-in effects refer to the finding that a wrong syntactic analysis becomes harder and harder to reanalyse the longer the ambiguous region is. As an example, consider the sentences in (9). Sentences (9-a) and (9-b) are initially ambiguous at the NP *the book* with respect to whether the NP is an argument of the verb *write* or the subject of the main phrase, while sentences (9-c) and (9-d) are not (because the verb already has an argument, *the essay*. Subjects initially interpret *the book* as an object of *write* because it is a semantically very likely object of *write* and because *write* is more often seen as a transitive verb than as an intransitive one.

It has been shown using acceptability judgements (Ferreira and Henderson, 1991) and reading times (Tabor and Hutchins, 2004), that (9-b) is much more difficult than (9-a) and the control condition (9-d). One would expect (9-b) to be a the most difficult condition anyway, as it is more complex than (9-b) and more ambiguous than (9-d). However, Tabor and Hutchins (2004) found a difficulty effect on the last word of (9-b) (an interaction between length and ambiguity before encountering *grew*) that goes beyond the main effects of ambiguity and complexity.

(9) a. As the author wrote the book grew. (ambiguous, short)
b. As the author wrote the book describing Babylon grew. (ambiguous, long)
c. As the author wrote the essay the book grew. (unambiguous, short)
d. As the author wrote the essay the book describing Babylon grew. (unambiguous, long)

Our model can predict both the ambiguity effect and the complexity effect, and would hence predict that (9-b) would be the most difficult sentence. However, it does not predict the effect to be super-additive, i.e. it does not predict the interaction between ambiguity and complexity found in the reading time experiments.

It is however conceivable, that the interaction effect is not a purely syntactic effect where the parser gets stuck with an analysis, but that the difficulty is due to semantic effects that enforce the object interpretation in (9-b). Our model trained on the Wall Street Journal does not reflect the fact that *book* is such a good object of *write* (*book* is only once seen as an object of *write* in the training material). For the model, sentence (9-b) is hence less difficult than for us, who know that book is a good object of write. Consider for example sentence (10) which has the same structure as sentence (9-b), but with the ambiguous noun phrase being a bad semantic object for the verb *write*.

(10) As the author wrote the wind coming from the east strengthened.

(We are not aware of this having been tested, but it seems likely that observed difficulty effects on the last word would not go beyond the additive effect of the ambiguity and complexity effects for sentence (10).)

9.1.8 Storage Costs

Storage cost (SC) is the second component of Dependency Locality Theory. Evidence for SC is reported in (Chen et al., 2005): if lots of open dependencies need to be kept in memory, processing is slower. This effect was found for verbal dependencies, wh-filler-gap dependencies and the expected PP argument for a verb. An example for the verbal dependencies is shown in (11). Reading times in the critical region, which is shown in italic print in (11), was slowest for the condition with two open verbal dependencies (11-a), next slowest was the condition with one long verb dependency (11-b), slightly faster was the sentence with one short verb dependency (11-c) and fastest the condition with no open verb dependency (11-d).

(11)
a. The detective suspected that the thief knew that *the guard protected the jewels* and so he reported immediately to the museum curator
b. The detective suspected that the knowledge that *the guard protected the jewels* came from an insider.
c. The suspicion that the thief knew that *the guard protected the jewels* worried the museum curator.
d. The suspicion that the knowledge that *the guard protected the jewels* came from an insider worried the museum curator.

The sentence processing theory presented here does not currently attribute difficulty to maintaining predictions in memory (even though such a component could be added easily, because the information which predictions have to be maintained for how long is readily available in the model). Neither the Surprisal nor the verification cost component can account for the storage effect. We originally decided against attributing difficulty to storing predictions because we were not able to find a significant effect of storage cost on the naturalistic data, when implementing Gibson's storage cost and evaluating it on the broad coverage Dundee Corpus.

9.1.9 Garden-Path Effects

While many of the effects discussed in this chapter so far are only detectable in reading times (and some of them in acceptability ratings), *garden path sentences* refer to a situation where a reader (or listener, although garden path sentences are more common in reading because intonation helps in finding the correct reading) becomes aware of the difficulty of the sentence, often initially judging it ungrammatical, even though the sentence is grammatical. The reader is initially "lured" into an initial very probable interpretation which later turns out to be incompatible with the end of the sentence.

The most famous example is probably Bever's (1970) sentence shown in (12):

(12) The horse raced past the barn fell.

The reader initially analyses *raced* as a simple past form and hence the main verb of the sentence. However, this is incompatible with *fell*. For the correct interpretation of the sentence, *raced past the barn* must be analysed as a reduced relative clause, and *raced* hence as a past participle.

Garden path effects are usually not only caused by difficult syntactic constructions, but are also dependent on semantics, i.e. they often consist of syntactically slightly difficult structures that in addition are made very implausible given the semantics of the sentence. For instance, Padó (2007) showed that difficulty ensues when syntactic and semantic interpretations are at odds. Our model so far does not account for these additional semantic effects, and therefore can't fully explain most garden path effects.

In order to explain the qualitative difference between a minor processing difficulty which people aren't even aware of and complete processing break-down, with some people not being able to recover the correct analysis at all, our model would assume that the correct analysis has fallen out of the reader's "search beam" because it was too unlikely when compared to alternative analyses. Hence, the sentence must be re-analysed from scratch, which is only successful if enough memory resources (in a human: concentration and cognitive abilities) can be made available for the larger beam needed to process the sentence. This is a standard account of explaining garden path sentences with a ranked parallel parser, which was already suggested in e.g. (Altmann and Steedman, 1988; Gibson, 1991).

9.1.10 Discussion

This section has evaluated, where possible, and otherwise discussed, the predictions of our prediction theory on nine psycholinguistic case studies. Our theory can simultaneously account for Surprisal effects like *either..or* prediction and facilitating ambiguity effects, and locality effects encountered in relative clauses (we have shown this for both psycholinguistic experimental material and naturally occurring relative clauses) and centre embedding. As our model is not implemented for German, it remains inconclusive, whether it can account for German anti-locality effects. Evaluation on English anti-locality effects did not reach significance, we therefore count them as 'not explained' (and will also count Levy's (2008) Surprisal predictions on the same case as 'not explained' in our comparison Table in Section 9.4, as it makes equivalent predictions).

Furthermore, we have argued that our theory can predict garden path effects and potentially digging-in effects if combined with a semantic model, and that it will be able to predict local coherence effects if a super-tagger (based on words only and no gold-standard POS tags) for non-prediction trees is added to the implementation.

Our model cannot currently account for storage cost effects, even though a cost function measuring the amount of predicted structure could be easily integrated with our current processing theory. We defer a detailed summary and comparison with other theories to Section 9.4.

As a general note, we would like to emphasise that testing experimental materials with the model trained on the Penn TreeBank can be problematic as the experimental materials often include unusual lexemes or rare constructions that were not seen during training on the Wall Street Journal texts. Not having encountered such events often enough can lead to inaccurate parses, and hence wrong difficulty estimates, or biased outcome due to slightly different distribution of some events in the WSJ as opposed to other text genres.

9.2 Broad Coverage Evaluation on the Dundee Corpus

This section evaluates the PLTAG implementation of the proposed sentence processing theory on the broad-coverage Dundee Corpus. All parameters (i.e., decay, timestamps, beam width) are the same as for the evaluation on case studies. Like for the evaluation of DLT integration cost and Surprisal, which was presented in Chapter 5, mixed effects models are used.

9.2.1 Data

Using the PLTAG parser, we were able to parse about 80% of the words in the Dundee Corpus. In the remaining cases, no analysis could be found within the parser's beam width. This is a bit lower coverage (presumably due to differences in text genre between the Wall Street Journal and the Independent) than reported for evaluation on section 23 of the Penn TreeBank (93% coverage of sentences). The distribution of difficulty predictions are slightly skewed, with a tail of rare, very large predicted difficulty values, see Figure 9.4.

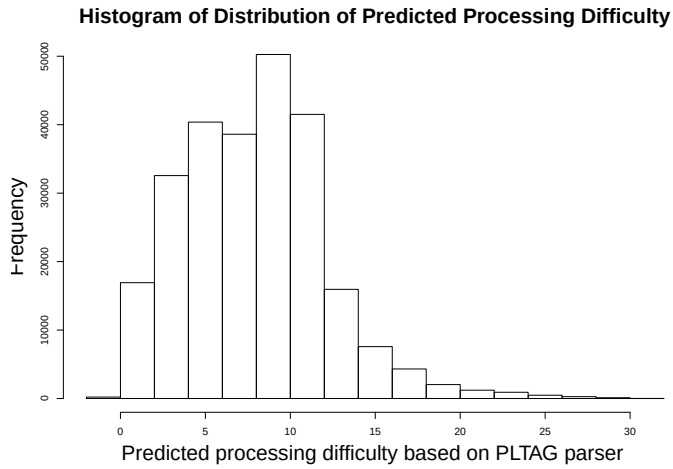


Figure 9.4: Distribution of PLTAG difficulty predictions.

9.2.2 Results

We evaluated our PLTAG-based model on the Dundee Corpus using the same model as for the broad-coverage evaluation of Surprisal and DLT integration cost, which included all predictors that we are not primarily interested in, as well as an intercept and their random slopes under subject. Outliers were removed using the > 3 standard

deviations in residuals criterion (cf. Section 3.2.5). We again use the term PREDICTIONTHEORY to refer to the explanatory variable for our model used in the regression models. PREDICTIONTHEORY is a significant positive predictor of reading times beyond what other factors included in the baseline model can explain. This is true for both first pass times and total reading times, see Table 9.2.

	First Fix		First Pass		Total Time	
Predictor	Coef	Sig	Coef	Sig	Coef	Sig
(INTERCEPT)	205.50	***	241.18	***	254.07	***
WORDLENGTH	0.71	*	8.11		7.36	***
WORDFREQUENCY	-6.33	***	-12.34	***	-15.80	***
PREVIOUSWORDFREQUENCY	-3.11		-6.19	*	-6.35	***
PREVIOUSWORDFIXATED	-10.95	***	-33.66	*	-35.60	***
LAUNCHDISTANCE	-1.63	***	-0.75		-0.86	
LANDINGPOSITION	8.31	***	-18.00		-21.39	***
SENTENCEPOSITION	-0.05	**	-0.24	***	-0.28	***
FORWARDTRANSITIONALPROB	-1.59	***	-1.97		-2.77	***
BACKWARDTRANSITIONALPROB	0.71	*	1.18		1.36	**
WORDLENGTH:WORDFREQUENCY	-1.15	***	-3.06	***	-4.15	***
WORDLENGTH:LANDINGPOSITION	rem	–	-19.21	***	-18.59	***
PREDICTIONTHEORY	0.09	*	0.20	**	0.33	***

$^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$

Table 9.2: Coefficients and significance levels for models of first fixation times, first pass durations, and total time for all words in the Dundee Corpus. The models include all predictors that are not of primary interest, interactions between them, and their slopes under subject. PREDICTIONTHEORY and its random slopes under subject were run on the residuals of the basic model. Predictors marked "rem" were removed from the regression because they did not significantly reduce the AIC.

Furthermore, there is a small significant effect on first fixation times. The PREDICTIONTHEORY effect on reading times seems very stable, we also find it in a simpler design regression model where PREDICTIONTHEORY is entered as a predictor without residualizing or fitting slopes under subject, and also when PREDICTIONTHEORY is used as an only predictor for reading times.

We can further analyse the difficulty predictions from the theory proposed in this

	First Pass		Total Time	
Predictor	Coef	Sig	Coef	Sig
PREDICTIONTHEORY-VERIFICATION	0.17		0.47	
PREDICTIONTHEORY-SURPRISAL	0.36	***	0.62	***

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 9.3: Coefficients and significance levels for the Surprisal and verification components of PREDICTIONTHEORY, regressed against the residuals of the main model from Figure 9.2.

work by taking a look at its two components, Surprisal and verification cost. We find that when fitted to the residuals of the model of other predictors, both of them have positive coefficients, but only the one for the Surprisal component reaches significance, see Table 9.3. As was the case for integration cost, the verification cost component assigns a cost of zero to many words.

9.2.3 Comparison to DLT and Surprisal

The predictions from our theory can be expected to be correlated with Surprisal calculated based on the Roark parser (see Section 5.5.1) and might have some correlation with DLT integration cost. The full table of correlations between different predictors of sentence processing difficulty is shown in Table 9.4. The strongest correlation exists between lexical Surprisal and the predictions by our theory, as expected, whereas the correlation with structural Surprisal is rather small. Furthermore, integration cost is more strongly correlated with our theory than with either of the Surprisal measures.

	INTEGRATION COST	LEXICAL SURPRISAL	STRUCTURAL SURPRISAL
LEXICALSURPRISAL	0.19		
STRUCTURALSURPRISAL	-0.09	0.36	
PREDICTIONTHEORY	0.26	0.53	0.10

Table 9.4: Correlation coefficients (Pearson's r) between the predictors, for fixated words ($N = 157,538$) that have been assigned a difficulty estimate by the prediction theory model.

The theory presented in this work, and implemented using PLTAG works better than Surprisal because PLTAG explains the data better: LEXICALSURPRISAL makes

wrong predictions, meaning that higher Surprisal would lead to faster reading. Similarly, we can argue that PREDICTIONTHEORY works better than INTEGRATIONCOST, as INTEGRATIONCOST doesn't make correct predictions on the broad-coverage data, see also discussion in Sections 5.3 and 5.4.

STRUCTURALSURPRISAL does make correct predictions, and turns out to improve the model similarly much as PREDICTIONTHEORY. To compare the two predictors, both predictors and their random slopes under subject were added into a first-pass-durations regression model. We then compared the model with structural surprisal and prediction theory to two separate models with only one of the factors. In both cases, there was a small but significant decrease of model fit, and the AIC score was exactly identical. Similar values were obtained from comparing total time models (the model including PREDICTIONTHEORY was one AIC count better than the STRUCTURALSURPRISAL model, but this clearly is not a significant difference). On these grounds we can't argue for either of the models over the other based on the broad coverage data alone.

A way in which the measures do differ is that PREDICTIONTHEORY aims to be a more complete measure than STRUCTURALSURPRISAL in that it does account for lexical frequency effects and integration effects, in addition to structural Surprisal effects. Evidence for this is provided by the fact that when added to a baseline model which only contains low-level parameters WORDLENGTH, PREVIOUSWORDFREQUENCY, PREVIOUSWORDFIXATED, LAUNCHDISTANCE, LANDINGPOSITION, SENTENCEPOSITION, WORDLENGTH:LANDINGPOSITION, and random intercept and slopes under subject (i.e. excluding WORDFREQUENCY, FORWARDTRANSITIONALPROBABILITY and BACKWARDTRANSITIONALPROBABILITY), PREDICTIONTHEORY can explain more of the variance, and leads to much better model fit (lower AIC and BIC, $p < 0.0001$) than adding only STRUCTURALSURPRISAL to such a model. The models are shown in Table 9.5.

To get a better intuitive impression of the explanatory power of the predictors, it is also informative to consider the simplest possible model, where the measure of interest is the only predictor of reading times. In such a model (response variable total reading times, subset of words that were not skipped), PREDICTIONTHEORY explains just over 2.2% of the variance in the data (reporting Adjusted R-squared), while LEXICALSURPRISAL accounts for 1.9% of the variance, INTEGRATIONCOST accounts for 0.2% of the variance and STRUCTURALSURPRISAL only for 0.0005% of the variance. (In comparison, the best single predictors of reading times, word length

Predictor	Coefficient	Significance
(INTERCEPT)	247.64	***
WORDLENGTH	17.65	***
PREVIOUSWORDFREQUENCY	-5.77	***
PREVIOUSWORDFIXATED	-33.99	***
LAUNCHDISTANCE	-0.69	
LANDINGPOSITION	-23.26	***
SENTENCEPOSITION	-0.62	***
WORDLENGTH: LANDINGPOSITION	-26.15	***
PREDICTIONTHEORY	0.85	***
STRUCTURALSURPRISAL	0.6	***

$*p < 0.05$, $**p < 0.01$, $***p < 0.001$

Table 9.5: First pass duration baseline model including only low-level predictors, and predictors STRUCTURALSURPRISAL and PREDICTIONTHEORY separately estimated on the residuals of the baseline model.

and word frequency, each account for a bit more than 6% of the variance in the data. Note however that the low overall level of R^2 values is not a big concern to us – it mainly reflect the fact that the data is very noisy.)

9.2.4 Discussion

Mixed effects analysis showed that difficulty predictions from the theory proposed in this dissertation can account for a significant proportion of the variance observed in reading time data. The effect seems to be mainly driven by the Surprisal component of the model, but the verification component also makes a small contribution in the right direction. Future work should explore effects found during the analysis of DLT integration cost in Chapter 5, affecting the processing of verbs in the presence of auxiliaries and the processing of compound nouns, as well as cost on more words. A straight-forward way to extend verification cost in the proposed model would be to assign costs for the retrieval of each integration node, instead of only the ones needed to be retrieved for verification. We leave this to future work.

When comparing broad-coverage predictions from the PLTAG-based implementation of our theory with other theories, our theory clearly outperforms DLT integration cost (estimated based on the MINIPAR parser (Lin, 1998)) and lexical Surprisal (es-

timated from the Roark parser (Roark, 2001a; Roark et al., 2009)). Furthermore, our theory can account for a larger proportion of the variance in the data than structural Surprisal. We therefore conclude that the theory proposed here makes useful predictions and has the largest explanatory power for naturally-occurring text among theories compared in this work.

9.3 General Discussion

Taking together the evidence from the psycholinguistic case studies and the broad coverage evaluation, the theory of prediction in human sentence processing presented in this thesis has been shown to have very good explanatory power for specific psycholinguistic phenomena such as Surprisal and locality effects, as well as difficulty encountered in naturally occurring broad-coverage text. As a last step, we compare our theory to alternative theories with respect to the phenomena discussed here, see Section 9.4.

Before proceeding to the model comparison, We would like to note that the predictions of our theory can be modulated by a number of parameters and design decisions. For some of these factors, it was possible to make informed decisions, while others were based on assumptions coming from different sources such as linguistic theories about what the elementary trees should look like etc, and sometimes it was necessary to guess or try out different parameters, for example for the size of the decay effect or beam widths for the parser. The difficulty predictions also strongly depend on the language model from the Wall Street Journal, which is a rather specialised part of natural English, with some otherwise frequent words appearing rarely and some words that are specific to financial affairs and economy being regarded as very frequent by the model even though they wouldn't be for the average human. The same also holds, to some extent, for syntactic structures. Furthermore, in order to calculate Surprisal we used the pruned probabilities, so this only constitutes an approximation to the full probability space. Finally, the model is also restricted by the complete lack of semantic or discourse context, which is not only a different factor for predicting processing difficulty, but also influences sentence processing in that the information from a semantic, discourse and world model could disambiguate syntactic structures and certainly helps humans to rule out many of the implausible analyses generated by the parser.

9.4 Comparison to Other Models

This section compares the explanatory power of the proposed sentence processing theory with the sentence processing theories discussed in Section 2.2, and the recently developed HHMM model, see below. Among these theories, our prediction theory is most similar to Surprisal and DLT integration cost.

The present model differs from Surprisal most importantly in that it contains the additional verification component. The Surprisal component of our model is quite similar to Surprisal calculated on Roark's PCFG parser, which is evident from their high correlation ($r = 0.60$, Pearson's product-moment correlation; note that this number differs from correlations shown in Table 9.4, as it refers only to the Surprisal component of our measure). Differences occur through the use of different formalisms, a right-corner transformed PCFG vs. PLTAG, as well as implementational aspects such as beam widths. As shown in Table 9.6, these differences allow our model to correctly account for the asymmetry in English relative clauses and centre embedding on top of what Surprisal can account for. Furthermore, the broad-coverage evaluation showed that our theory is better suited for explaining difficulty in naturally-occurring text. Others have also evaluated Surprisal on the Dundee Corpus (Kennedy et al., 2003), Potsdam Sentence Corpus (Kliegl et al., 2006) and MIT narratives by Asaf Bachrach (Roark et al., 2009), but only structural Surprisal estimates, not lexical Surprisal, have so far been shown to correctly predict broad-coverage reading times.

The relationship between our theory's verification component and DLT integration cost is a bit more complex. DLT integration cost predicts processing difficulty based on the number of discourse referents that occur between the head and its preceding dependent(s). The relationship between the difficulty predictions and reading times is for simplicity suggested to be linear Gibson (2000), whereas verification cost in the model suggested here depends on the probability of the prediction tree that's verified, as well as the the distance of the initial prediction point measured in words and the decay factor, and modulated further by reactivations through integration of other words into the predicted structure. The correlation between DLT integration cost estimations and the verification cost component of our theory is $r = 0.27$ (according to Pearson's product-moment correlation; evaluated only on those words of the Dundee Corpus on which both theories make non-zero predictions). Our model can explain prediction effects like *either..or* processing, facilitating ambiguity and broad-coverage processing on top of what DLT integration cost can explain, see Table 9.6.

Theories	EOP	ERC	EAL	GRC	FA	LC	DI	SC	CE	BC
this work	+	+	(–)	?	+	?	?	–	+	+
Surprisal	+	–	(–)	+	+	–	–	–	NA	(+)
DLT	–	+	–	–	–	NA	NA	+	+	–
Memory&Activ.	NA	+	NA	+	NA	NA	NA	+	+	NA
Entropy	NA	NA	NA	NA	NA	NA	NA	NA	NA	+
Competition	NA	NA	NA	NA	(–)	NA	+	NA	NA	NA
HHMM	NA	NA	NA	NA	NA	NA	NA	NA	NA	+

Table 9.6: Comparing explanatory power of different sentence processing theories for empirical processing difficulty phenomena. EOP = Either...or prediction effects; ERC = classical English SRC / ORC asymmetry; EAL = English anti-locality effect (Jaeger et al., 2010); GRC = Anti-locality effects in German Relative Clauses; FA = Facilitating Ambiguity, LC = Local Coherence Effects; DI = Digging-In Effects; SC = Storage Cost Effects; CE = centre embedding; BC = broad coverage; '+' means that the effect can be explained, '–' means it cannot be explained, '?' means that it can in principle be explained (e.g. with the addition of a semantic model) but remains to be shown, and 'NA' means that, to the best of our knowledge, this effect has not been tested for that theory. For '(–)' refer to Sections 9.1.5 and 9.4.

Lewis and Vasishth's (2005) Memory and Activation model (see Section 2.2.8) is in some ways similar to DLT integration cost, but uses a well-motivated psycholinguistic account of human memory and activation of lexical items, which allows it to explain German anti-locality effects on top of what DLT can explain. It would be interesting to integrate the psycholinguistically motivated architecture for memory access into our model.

Entropy and Competition models are more different from the model proposed here, see Sections 2.2.6 and 2.2.7. Entropy has only been tested on broad-coverage text, but not on any of the phenomena evaluated here (as far as we are aware). The competition model is the only one among models compared here that can account for digging-in effects, see Table 9.6. Competition models have also been claimed to account for facilitating ambiguity (Green and Mitchell, 2006), on the ground that some people would choose the one analysis, and other the other one, such that for the unambiguous cases, some penalty for maintaining the other analysis would be incurred, but not in the ambiguous case. However, under this interpretation, a large variance between reading

times would be expected for the unambiguous condition, which has not been shown to be the case. Due to this controversy, the field is marked as '(–)' for Competition models in Table 9.6.

Concurrently to the model presented in this thesis, Schuler et al. (2008) have suggested a model which is motivated by incremental (though not fully connected), time-linear and memory-restricted processing. Their parser uses a hierarchical HMM which is right-corner transformed. The HHMM process itself however, does not seem to be a psycholinguistically well-motivated model. Parsing with the HHMM is not lexicalized. Schuler et al. (2008) found that a stack size of four is generally sufficient for parsing most of the Penn TreeBank. This is comparable to our results in that instead of using a stack, we connect fragments using prediction trees, and found that no more than 5 prediction trees (and in most cases, 4 prediction trees) are needed to parse the PTB. Wu et al. (2010) derive psycholinguistic measures from their HHMM parser by calculating Surprisal and counting average stack depth across parallel analyses for each word, which bears similarity with DLT storage cost. Both of these measures were found to be positive predictors in a regression model modelling self-paced reading durations on a corpus of four short narratives designed to contain large integration costs, by Asaf Bachrach. This is the same corpus used in this work to evaluate the implementation of DLT integration cost, see Section 5.3. This text is less well-suited for evaluating a model of sentence processing than the Dundee Corpus as it does not contain naturally occurring text, is more difficult to parse with a parser trained on newspaper (see Section 5.3), and uses self-paced reading (see critical comments in Section 2.1.4.1) instead of eye-tracking as a measure of processing difficulty. Difficulty predictions from the HHMM model have not been evaluated on any experimental data, see Table 9.6.

9.5 Conclusions

This chapter has evaluated the predictions of the processing difficulty proposed in Chapter 6, based on the implementation of the strictly incremental predictive parser described in Chapter 8 and based on the PLTAG formalism, as suggested in Chapter 7.

We found that our theory can simultaneously account for more of the empirical data than alternative theories, in particular, it correctly predicts both Surprisal and locality effects. Additionally, we compared our theory on broad-coverage text from the Dundee Corpus, replicating the studies presented in Chapters 4 and 5 with our theory as an

explanatory variable. We were able to show that our theory predicts reading times on the embedded verb of subject and object relative clauses better than either Surprisal or DLT integration cost, and that it can also correctly account for processing difficulty across the whole of the Dundee Corpus. We also showed that it was able to predict a larger proportion of the variance in the reading time data on the Dundee Corpus than either Surprisal or DLT.

In conclusion, we find a wide range of empirical support for our PLTAG-based theory of prediction and verification in human sentence processing, and show that it has larger explanatory power than previous theories of sentence processing.

Chapter 10

Conclusions and Future Work

This Chapter briefly summarizes the main contributions that this thesis makes, and points out directions for further research.

The first claim of this thesis, that the evaluation of psycholinguistic theories on broad-coverage data can be a valid additional resource to traditional lab experiments and that it can provide insights which cannot be obtained from traditional experiment data, was shown in Chapters 4, 5 and 9.2: Chapter 4 showed that an established effect, the subject vs. object relative clause asymmetry, can also be shown for relative clauses from naturally occurring text. Chapter 5 compared two existing theories, Dependency Locality Theory (DLT) and Surprisal on the broad-coverage corpus and found that DLT, while making correct predictions for verbs and some nouns, cannot correctly account for the difficulty effect across all words. The corpus study furthermore indicated that the role of auxiliaries and compound nouns needs to be investigated in more detail. Section 9.2 provides supporting evidence for the theory proposed in this work, in addition to the evaluation on case studies, and compares sentence processing theories also with respect to how much of the variance in the reading time data they can account for.

The second claim, stating that Surprisal and Dependency Locality Theory explain different aspects of sentence processing was shown in Chapter 5, where we found that both theories can explain some of the data, but their predictions are not significantly correlated.

The third claim was that modelling prediction and verification processes is cognitively plausible, and that it provides a framework for combining aspects from DLT and Surprisal. The plausibility of prediction and verification is supported by the discussion of recent experimental evidence in Chapter 6. We have also shown that our model

incorporating the explicit prediction and verification mechanism is very successful at predicting a range of experimental data (Section 9). Finally, Chapter 8.7 showed how the modelling of prediction and verification naturally combines aspects of Surprisal and Dependency Locality Theory.

10.1 Main Contributions

The first part of this thesis showed that eye-tracking data from naturally occurring text can be a beneficial complimentary method for evaluating theories of human language processing. The second part developed, implemented and evaluated a model of prediction coupled with a verification process for human sentence processing. The results provide some methodological, experimental and theoretical contributions, as well as the PLTAG version of the Penn Treebank as a resource and a strictly incremental parser for TAG as an NLP tool.

- Evaluation on a broad-coverage corpus.
  Evaluation on broad-coverage text allows to detect processing difficulty on structures in context and possibly tease apart effects introduced by the experiment from processing difficulty encountered in every-day processing. Furthermore, theories of sentence processing can be tested as to whether they scale up to broad-coverage, naturally occurring text, an aspect which is particularly relevant with respect to applying psycholinguistic theories for sentence processing in NLP applications.
  Demonstrating the usefulness of eye-tracking corpora will hopefully motivate the creation of similar corpora that overcome some of the limitations of the Dundee Corpus, in particular the fact that no manual parse tree annotation is available for the Dundee Corpus, and that it was only read by 10 subjects.

- Demonstration that syntactic processing effects can be found in eye-tracking data of naturally-occurring text.
  Previous models of reading primarily focussed on low-level reading effects. We have shown in our studies in Chapters 4 and 5 that higher-level syntactic effects that can also be detected in the reading data, and that they help to account for a part of the variance in the reading data that lower-level effects cannot account for.

- Comparison of alternative sentence processing theories on the same resource.
  Three theories of sentence processing, DLT integration cost, Surprisal and the prediction theory proposed in this thesis were evaluated on the same resource. This made it possible to compare effect sizes in terms of variance accounted for, and determine whether the same or different parts of the variance were explained by the theories.

- Implementation of DLT.
  To the best of our knowledge, this work performed the first implementation of DLT. Before, DLT had been calculated by hand for a small number of materials used in experiments.

- Definition and Formalization of PLTAG.
  The design of PLTAG was guided by the principles of incrementality and connectedness and includes an explicit mechanism for generating and verifying syntactic predictions. Chapter 7.2 argued that PLTAG and LTAG are strongly equivalent. PLTAG differs from the most similar TAG variant, DVTAG, in that its lexicon is much smaller, providing adaptable prediction grain size, and having shown that parsing with it is tractable.

- PLTAG Treebank and Lexicon.
  We converted the Penn TreeBank into PLTAG format and induced a PLTAG lexicon, consisting of canonical LTAG trees and prediction trees.

- Proposition of a new cognitively plausible theory for syntactic processing.
  The theory proposed in this work is based on psycholinguistically plausible assumptions including incremental processing, memory decay and an explicit prediction and verification mechanism.

- Implementation of a strictly incremental statistical parser for PLTAG.
  The parser is to our knowledge the first strictly incremental predictive parser. We show that it can tractably parse broad-coverage data and that it achieves accuracy results that make it suitable for using as a basis for evaluating the sentence processing theory.

- Implementation of the PLTAG-based sentence processing theory.
  The full implementation of the proposed theory makes it possible to evaluate predictions for processing difficulty automatically on both experimental items

and broad coverage text. Furthermore, it can make predictions on untested phenomena that can then be verified in a laboratory experiment.

- Evaluation of the PLTAG-based sentence processing theory.
  The model of sentence processing difficulty proposed in this thesis 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. Furthermore, it is validated by the broad-coverage evaluation: it can explain the variance in reading times on naturally-occurring text better than alternative models like Surprisal and DLT integration cost. Our model therefore constitutes an important step towards a unified theory of human parsing.

10.2 Directions for Further Research

The results obtained in this thesis open directions for further research. This section point outs some interesting future directions which build on the present work.

10.2.1 Evaluation of Theory in Other Languages

The theory developed in this thesis was only implemented and tested on English. However, a plausible theory of human sentence comprehension should make correct predictions for a range of (ideally all) languages. PLTAG could also be used as a formalism for implementing our theory in other languages, because it is mildly context-sensitive and can therefore model cross-serial dependency constructions that have been argued to exist in Dutch (Bresnan et al., 1982) and Swiss German (Shieber, 1985).

Studying crosslinguistic phenomena is particularly interesting because some psycholinguistic effects may only be tested based on manipulations that are not grammatical in English, but could be manipulated and tested in another language where they are grammatical.

In order for our theory to generalize to other languages, it must also take into account aspects of language that are currently largely ignored, such as morphology. A first starting point would be to train the parser and test the theory on French, because there is a broad-coverage French eye-tracking corpus available, which was also collected by the Dundee group (Kennedy and Pynte, 2005). An interesting aspect in which French differs from English is for example headedness of noun compounds (in

English, the head is usually the last component, whereas in French it tends to be the first component).

10.2.2 Integration with Semantics and Discourse

In recent years, people have argued that the traditional view on language processing is often too syntacto-centric (Jackendoff, 2003; van Berkum et al., 2007). This thesis focussed only on syntax, but this is of course far from the whole story. We saw in Section 9.2 that the syntactic predictors indeed only explain a small proportion of the variance in the reading data. A first step towards combining syntactic and semantic effects into a combined Surprisal measure have been undertaken in (Mitchell et al., 2010), where we combine syntactic Surprisal based on the Roark parser with semantic Surprisal (based on LSA) and calculate a combined Surprisal measure that improves model fit over just syntactic Surprisal. Ideally, however, we would like to use the parser presented here, as we have shown that surprisal estimates from the PLTAG parser match reading times on the Dundee corpus better than surprisal estimates based on the Roark parser, and because it implements a verification component. The semantic and syntactic components should also be integrated better, such that they can inform each other: the semantic component should take into account syntactic relationships, and the parser could take semantic plausibility into account when rating analyses. LSA only quantifies how realted two lexemes are irrespective of their roles to each other or in the sentence. A deeper semantic approach could calculate semantic analyses based on the TAG derivations. Modelling discourse effects, reference resolution and etc. would be important further steps towards a more holistic model of human language processing.

10.2.3 Incremental Update of the Language Model

The model proposed in this thesis only deals with language comprehension, and is agnostic as to how the probabilities are acquired, and to how they would change over time (in fact, in the current training of the model, probabilities are based on the batch of events observed during training, and then never updated). A better model would take into account short term priming as well as long term learning from observing new events during run time. Dubey et al. (2006) has shown how a probability model for a parser can be updated during parsing to model short term priming. Structures which have been encountered recently are then predicted to be easier and hence faster to process than otherwise expected. Over time, this effect of exposure to a specific

encountered structure weakens, but will have a small effect on the overall probability distribution in the model. Such learning effects have also received empirical support from a recent study (Wells et al., 2009) that exposed several groups of adults to different stimuli over a couple of weeks, and showed that it affected how difficult they found the processing of such structures at the end.

10.2.4 Experiments on Prediction Grain Size

The specification of the sentence processing theory and the PLTAG formalism in Section 7.3 raised questions about the exact grain size of predictions. Are arguments principally different from modifiers in that they are predicted while modifiers are not? And which role does context play in this respect? It has been shown that modifiers can be predicted if they are required to semantically disambiguate a referent. How detailed are the predictions? How specific are predictions of subcategorization frames, and when are they generated? And how does the prediction of subcategorization frames generalize to languages with free word order?

These questions should be answered based on laboratory experiments.

10.2.5 Automatic Determination of Lexicon Entry Sizes

The extended domain of locality in TAG gives us the flexibility to generate lexicon entries with more than one lexical anchor, and thus model lexicalized multi-word expressions and idioms. As pointed out in Sections 6.1.2 and 7.3.2, it would be desirable to learn in a more principled fashion from data what size a lexicon entry should be, for example using a DOP-like framework or extending the leaning process proposed in (Cohn et al., 2009) for Tree Substitution Grammar to PLTAG structures.

10.2.6 Modelling the Effect of Processing Difficulty on Reading Times

As briefly discussed in Chapters 2 and 9, theories of sentence processing try to predict processing difficulty, and evaluate predictions most often on reading times. If done on a word-by-word basis, like in our broad-coverage evaluations in Chapters 5 and 9.2, possible spill-over and skipping effects are not accounted for. It is possible that this severely affects some words, like e.g. the determiner of an unexpected noun phrase, which will receive a high difficulty prediction by Surprisal-based theories, which is

most likely to be only empirically measurable on the following noun due to common skipping of words that are short and frequent. One way of solving this issue would be to integrate the difficulty predictions into a model of eye-movement, such as the ones discussed in 2.1.3, in particular E-Z Reader 10, which suggests a mechanism for integrating higher-level processes (Reichle et al., 2009), or Klinton Bicknell's recent rational model of reading (Bicknell and Levy, 2010).

In a similar line or argument, a better mathematical model needs to be developed in order to include information about skipping into the regression models.

10.2.7 Improving Parser Performance

Finally, the performance of the incremental predictive PLTAG parser presented in Chapter 8 falls short of the parsing accuracy achieved by today's state-of-the-art parsers. It is however very likely, that performance can be substantially improved if some psycholinguistically motivated constraints are relaxed. First promising steps would be to introduce supertagging for canonical trees in addition to prediction trees. In addition, the lexicon could be changed to be smaller, thus reducing data sparseness, by leaving out all traces and doing sister adjunction (Chiang, 2000) instead of standard TAG adjunction.

Appendix A

CCG and Incrementality

Combinatory Categorial Grammar (CCG, Steedman, 1996, 2000), is a grammatical theory which provides a transparent interface between surface syntax and underlying semantics. Each (complete or partial) syntactic derivation corresponds directly to an interpretable structure. This allows CCG to provide an account for the incremental nature of human language processing. As we will discuss in this chapter, CCG with the standard lexicon and rules does however not always allow for the strongest interpretation of incrementality.

In CCG, the language-specific knowledge about the grammar is stored in the lexicon. There is a finite set of rules that allow the lexical categories to combine. These rules are based on the categorial calculus (Ajdukiewicz, 1935; Bar-Hillel, 1953) as well as on the combinatory logic of Curry and Feys (1958).

This section is first going to give an overview of the CCG combination rules, before discussing incremental processing in CCG. Most examples used for explaining CCG rules are taken from Steedman (2000), which the reader should also refer to for further detail.

A.1 Standard CCG Rules

Each word in the lexicon is assigned one or more categories that define its behaviour in the sentence. Categories for a word can either be atomic e.g., NP, S, PP, or complex like the category $(S\backslash NP)/NP$. Complex categories X/Y and $X\backslash Y$ designate a functor-argument relationship between X and Y, where the directionality of the relation is indicated by the forward slash $/$ and the backward slash $\backslash$. For example, category X/Y takes category Y as an argument to its right and yields category X, while category $X\backslash Y$

takes category Y as an argument to its left to result in category X.

These two rules are referred to as forward and backward functional application, shown in Rules (1-a) and (1-b).

(1) *Functional Application*

a. $X / Y \quad Y \quad \Rightarrow \quad X \quad (>)$

b. $Y \quad X \setminus Y \quad \Rightarrow \quad X \quad (<)$

Figure A.1 shows natural language examples for functional application. With only functional application, it is possible to derive normal form parses of traditional constituents of English sentences that do not involve any kind of traces or movement.

(a) forward application (b) backward application

Figure A.1: Examples of forward and backward application in CCG.

In addition to these two most basic operators, the canonical CCG inventory as defined in (Steedman, 2000) contains a range of further operators. Forward and backward composition for example are needed to allow for non-standard constituents, such as in *"Mary [[bought] and **[will eat]** carrots."*. Essentially, composition allows to apply a functor to its argument even if that argument is a functor itself, i.e. if the argument has dependents itself. The simplest case of this are Forward and Backward Composition, see Rules (2) and Figure A.2, but CCG allows the same also for cases where multiple dependents are missing (Generalized Composition), as in *"John [[recommended], and **[will give]**, a book to Mary."*, see Rules (3) and Figure A.3. Composition is used when the argument of a function is itself still expecting arguments. These expected arguments are then inherited to the functor category.

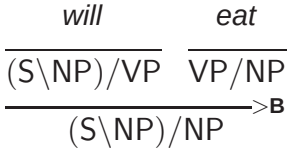


Figure A.2: An example of forward composition in CCG.

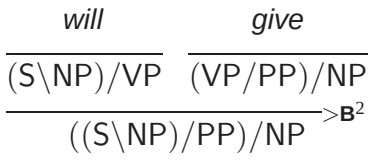


Figure A.3: Examples of generalized forward composition in CCG.

(2) *Composition*

a.	X / Y	Y / Z	⇒	X / Z	($>B$)
b.	Y \ Z	X \ Y	⇒	X \ Z	($<B$)

(3) *Generalized Composition*

a.	X / Y	(Y / Z) / $1	⇒	(X / Z) / $1	($>B^n$)
b.	(Y \ Z) \ $1	X \ Y	⇒	(X \ Z) \$1	($<B^n$)

CCG also has unary rules, known as type-raising rules, see Rules (4) and Figure A.4. Type-raising turns an argument of a function into a function which takes the original function into its argument. Type-raising usually occurs together with composition and is necessary for filling argument slots of the functor that are not directly accessible. The argument can turn into a function through type-raising and then inherit the remaining arguments of the original function through composition. Type-raising is often used for subjects, e.g. in sentences like *[[**Peter bought**] and [**Mary ate**]] the carrot.* To prevent extensive type-raising that would potentially lead to over-generation of the grammar, T \ X and T / X have to be licensed categories for the language.

(4) *Type-raising*

a.	X	⇒	T / (T \ X)	($>T$)
b.	X	⇒	T \ (T / X)	($<T$)

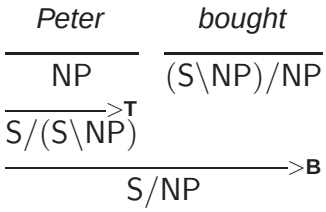


Figure A.4: Example of forward type raising in CCG.

The intuition behind type-raising of NPs is that the argument can by virtue of its

case demand for a specific predicate in order to build a proposition. In inflecting languages, a dative NP can therefore select for different verbs than a accusative NP. Nominative vs. accusative NPs would just be type-raised to different complex categories, reflecting their function in the sentence.

Examples for type-raising English NPs in their different functions occurs for example, when combining clusters of English NPs into constituents (for example for a clause such as *"give [a teacher an apple] and [a policeman a flower]"*, see Figure A.5).

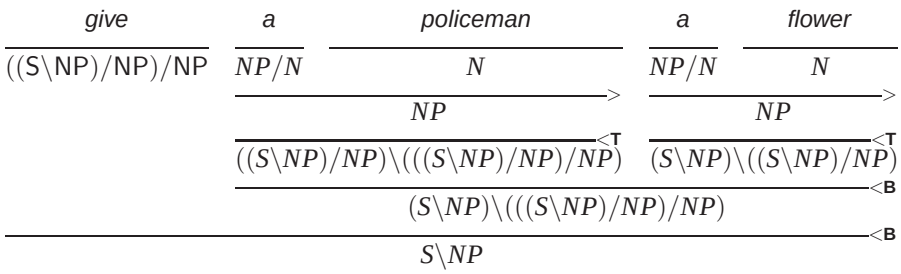


Figure A.5: Example of type raising with type-raising categories for NPs in different syntactic functions.

Type raising and composition rules are also necessary to parse extractions like in relative clauses, where, just as in coordination, the nearest argument is not directly available.

The last of the standard CCG rules is forward and backward (crossed) substitution. Substitution rules are needed for so-called "parasitic gaps", extracted items with more than one dependency, as in *"articles which$_i$ I file$_i$ without reading$_i$"*.

(5) *Substitution*

a. Y / Z (X \ Y) / Z $\Rightarrow$ X / Z ($<S_\times$) where Y = S \ $

Finally, there is a special ternary coordination rule shown in Rule (6). This rule (and the associated lexicon entry *CONJ* for "and" / "or" etc.) is preferable[1] over the lexicon entry of a conjunct $(X\backslash X)/X$.

(6) *Coordination*

a. X CONJ X' $\Rightarrow$ X" ($<\Phi>$)

[1]The $(X\backslash X)/X$ rule allowed for problematic application of unary rules or composition to the conjuncts before coordination was completed, see Steedman (2000).

The full set of CCG rules includes crossed versions of forward and backward composition and substitution. However, it depends on the language whether how many of these rules are allowed to be applied. In English, forward crossed composition is forbidden, because it would cause over-generation. Generally, crossed rules are "dangerous" in order-sensitive languages, because they can lead to accepting ungrammatical word sequences. On the other hand, English does permit backward crossed composition. This rule is needed in order to account for heavy NP-shift. To prevent over-generation, both of the backward crossed rules (composition and substitution) are restricted in what categories they can be used on, as seen in Rule (5).

A.2 The limits of CCG and Over-generation

CCG rules create so-called spurious ambiguity. This means that there are alternative ways and orders of applying these rules, which lead to syntactically distinct but semantically equivalent derivations of a sequence of words. The combination of type-raising and composition can be used to construct almost any syntactic tree for a sequence of words. Current CCG parsers create CCG normal form derivations, which means that they use type raising and composition only when the sentence can't be correctly parsed otherwise. However, the possibility of building non-standard of constituents, which are licensed by how phrases can be coordinated, allows CCG to make more incremental derivations and thereby explain some of the incrementality that is observed in the human parser.

However, it is not possible to build an arbitrary derivation using these rules, and this sets a limit on how incremental a bottom-up CCG parser with the standard rules can actually be. For example, the standard set of rules is not sufficient to build an incremental derivation of object relative clauses like *reporter who John attacked*, even though it is grammatically possible to form a coordination of the form: *[[the reporter who John] and [the senator who Mary]] attacked admitted the error*. The normal form derivation of an object relative clause is shown in Figure A.6. In order to parse the coordinated phrase, it is necessary to use an additional rule, called *Geach Rule*. Geach rule is not part of standard CCG. The normal form derivation of the coordinated ORCs with extracted verbs is shown in Figure A.7. The Geach rule is a unary rule, see Rule (7). Interestingly, composition can be reduced to Geach rule and functional application (see Figure A.8). (This is where the $>$ **B** notation for composition comes from.) Similarly, there's a unary version of substitution, see Figure A.9.

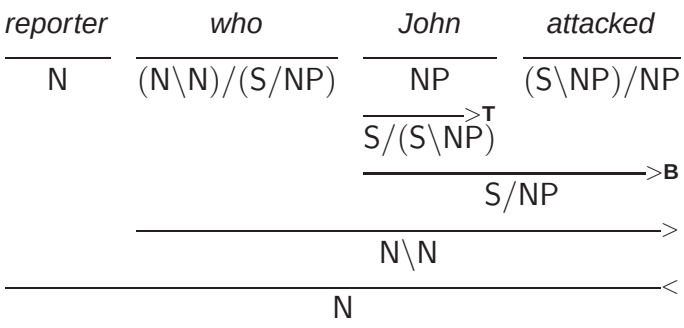


Figure A.6: Example of normal form derivation for object relative clause in CCG.

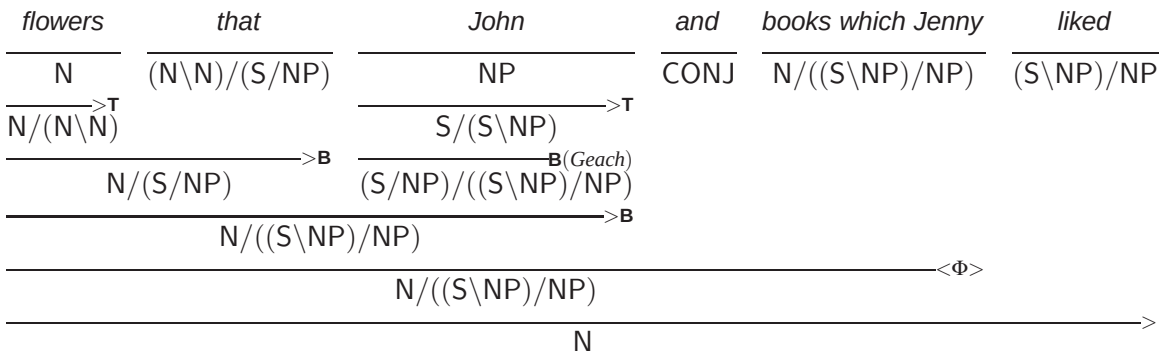


Figure A.7: Example of incrementalised derivation for object relative clause in CCG (coordination is not incremental in this derivation).

(7) *Geach*

a. $Y / Z \quad \Rightarrow \quad (Y / G) / (Z / G) \quad (B)$

This means that the Geach rule actually *is* technically part of traditional CCG, but it usually only occurs wrapped up with functional application. But what happens, if we want to take incrementality a step further and try to substitute the ORC noun ("John" in Figure A.7) by an NP like "every accordionist"? CCG cannot parse object relative clauses which contain an embedded NP of length greater one fully incrementally, see the derivation in Figure A.10, where the word *man* cannot be integrated with the sentence prefix.

Whether CCG should be able to strictly incrementally derive object relative clauses is however open to discussion, as it depends on whether a sentence such as *"[[books that every] and [journals that no]] accordionist liked"* is judged as similarly good and grammatical as *"flowers that John and books which Jerry liked"*, and whether it can actually be shown that human processing is strictly incremental at this point.

The most incremental analysis of an object relative clause is not fully incremental

$$\frac{X/Y \quad Y/Z}{X/Z}\,{>}\mathbf{B}$$

(a) functional composition

$$\dfrac{\dfrac{X/Y \qquad Y/Z}{(X/Z)/(Y/Z)}\,\mathbf{B}}{X/Z}\,{>}$$

(b) Geach and functional application

$$\frac{X/Y \quad (Y/Z)/W}{(X/Z)/W}\,{>}\mathbf{B}^2$$

(c) generalized composition

$$\dfrac{\dfrac{\dfrac{X/Y \qquad (Y/Z)/W}{(X/Z)/(Y/Z)}\,\mathbf{B}}{((X/Z)/W)/((Y/Z)/W)}\,\mathbf{B}}{(X/Z)/W}\,{>}$$

(d) Geach twice and functional application

Figure A.8: Functional composition (first derivation) can alternatively be understood as a combination of the Geach rule and functional application (second derivation).

$$\frac{Y/Z \quad (X\backslash Y)/Z}{X/Z}\,{>}\mathbf{S}$$

(a) Binary substitution

$$\dfrac{\dfrac{Y/Z \qquad (X\backslash Y)/Z}{(X/Z)/((X\backslash Y)/Z)}\,\mathbf{S}}{X/Z}\,{>}$$

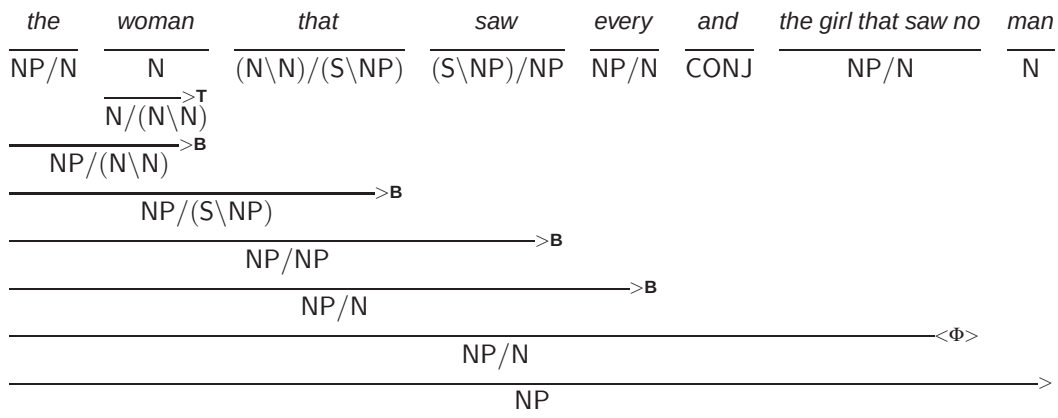
(b) Unary substitution and functional application

Figure A.9: Binary substitution can be decomposed into unary substitution and functional application.

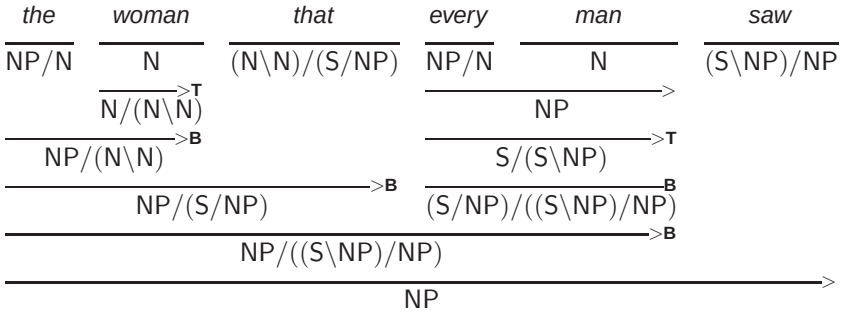
(see Figure A.11 (b)), since one can only type-raise the NP "every man" once the words *every* and *man* they have been combined. Note though that the determiner can be integrated incrementally into other prefixes (e.g. in subject relative clauses), as shown in the first derivation in Figure A.11 (a). Grammaticality is a gradual process, but it

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\textit{books}}{N}}{N/(N\backslash N)}\,{>}\mathbf{T} \quad \dfrac{\textit{which}}{(N\backslash N)/(S/NP)}}{N/(S/NP)}\,{>}\mathbf{B} \quad \dfrac{\dfrac{\dfrac{\textit{every}}{NP/N}}{S/(S\backslash(NP/N))}\,{>}\mathbf{T} \quad \dfrac{\textit{and}}{CONJ} \quad \dfrac{\textit{journals that no}}{N/((S\backslash(NP/N))/NP)} \quad \dfrac{\textit{man}}{N}}{(S/NP)/((S\backslash(NP/N))/NP)}\,{>}\mathbf{B}}{N/((S\backslash(NP/N))/NP)}\,{<}\Phi{>}$$

Figure A.10: Example of incrementalised derivation for object relative clause in CCG (coordination is not incremental in this derivation).



(a) Subject relative clauses can be parsed incrementally, even though a wh-constraint is violated.



(b) Object relative clauses cannot be parsed incrementally because of the subject island constraint (the NP has to be type-raised as a whole).

Figure A.11: Incrementality in subject relative clauses, which, contrarily to the object relative clause, does get a derivation in CCG.

does not seem obvious why “Here come the woman that saw every and the girl that saw no man.” would be much more grammatical than “Here come the woman that every and the girl that no man saw.”, and thus justify why the one can be derived in CCG and the other one can’t.

A.3 Incrementality in CCG

This section further analyses CCGs failure to fully incrementally derive object relative clauses. Table A.1 lists all possible category constellations for a sequence of three words, which are functors and arguments of one another. There are five constellations which only use composition and functional application.

The other tree constellations (6., 7. and 8. in Figure A.1 require type-raising. De-

1.	a/b	b/c	c				
2.	(a/c)/b	b	c				
3.	a/b	b	c\a				
4.	a	(b/c)\a	c				
5.	a	b\a	c\b				
6.	a	b	(c\a)\b	⇒	c/(c\a)	(c\a)/((c\a)\b)	(c\a)\b
7.	a/c	b	c\b	⇒	a/c	c/(c\b)	c\b
8.	a	b/c	c\a	⇒	b/(b\a)	(b\a)/(c\a)	c\a

Table A.1: Category constellations in a sequence of three adjacent words that are functors and arguments of one another.

pending on the parametrisation of a specific language, not all of the type-raising rules for 6. and 7. would be parametrically licensed in standard CCG, which means that it depends on the specific instance whether a sequence of categories is parsable incrementally. In the eighth case, the functor $c\backslash a$ is not directly adjacent to its argument a. Instead, there is another word is another word in the middle which takes c as its argument. These categories can still be combined incrementally using type-raising and geaching, but the type-raising required for this kind of operation would likely not be licensed by the language (since there's no category that subcategorises for its grandchild).

But what happens in CCG categories which cannot be parsed incrementally, even when not following the type-raising restrictions, such as object relative clauses with an NP that's composed of a determiner and a noun? This case requires a constellation of four categories:

a/(s/np)	np/n	n	(s\np)/np	⇒
a/(s/np)	(s/np)/((s\(np/n))/np)	n	(s\np)/np	

Even after type-raising it is not possible to process this category constellation incrementally, due to the first category of the accusative relative pronoun category. There are however ways around this problem. For example, the category of the object relative pronoun could be changed. If the category was $(a/((s\backslash np_i)/np))/np_i$ instead of a/(s/np), an incremental derivation would be unproblematic.

However, there are theoretically motivated reasons for the original object relative pronoun category (N\N/(S/NP)): the standard category prevents subject island violations by burying the subject NP in the verb category instead of including it in the

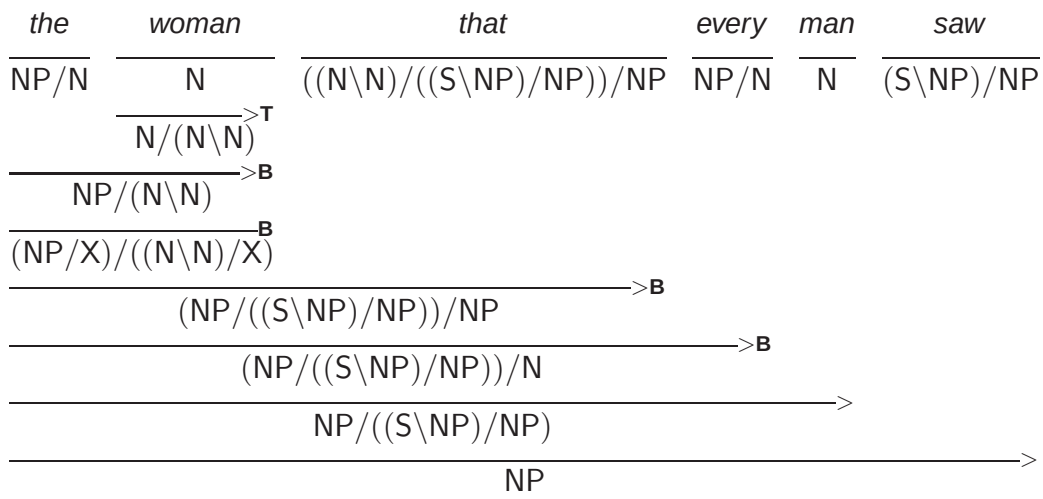
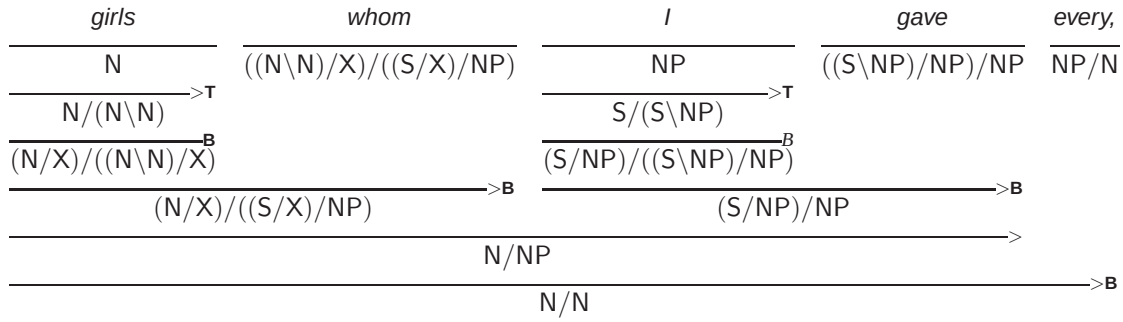


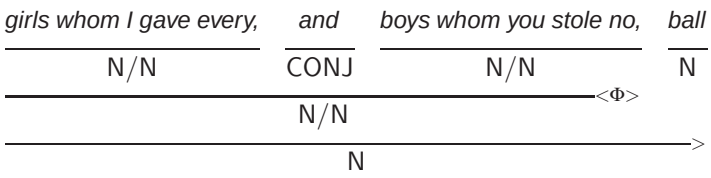
Figure A.12: Incremental derivation of object relative clause with new object relative pronoun category.

category of the object relative pronoun. (The NP argument in (N\N/(S/NP)) is for the object NP.) The subject NP is thus not accessible from outside the relative clause. This property is interesting, and raises the question whether it can be used in a directed manner in order to enforce island constraints. For example, the wh-island constraint is violated in subject relative clauses like *the man that every and the woman that no kid saw*, and there are also ungrammatical object relative clauses with a dative as a relative pronoun, where the ungrammaticality is due to the extraction of parts of the object from within the relative clause. In this case, the category of the dative relative pronoun does not prevent extraction to outside the NP and CCG therefore over-generates in these cases: *"[girls whom I gave every] and [boys whom you stole no], ball"*. See Figure A.13 for the complete derivation of this sentence. See Baldridge (2002) for more examples of CCG over-generating on relative clauses. So the change of the object relative clause category to the more complex category seems defensible, given that it would allow fully incremental derivations of object relative clauses, and lead to less of an asymmetry in terms of when island constraints are violated in CCG.

To summarise this observation, we conclude that incremental processing can't be guaranteed to be possible if the categories are not "deep enough" to reflect the structure of the relevant part of the tree. The more complex version of the direct object relative pronoun category $((N\backslash N)/((S\backslash NP_{subj})/NP_{obj}))/NP_{subj}$ reflects that a subject NP is required first, followed by a transitive verb to the right, while the standard simple object relative pronoun category $(N\backslash N)/(S/NP_{obj})$ only encodes that a sentence lacking an object NP is needed, but does not encode the subject NP. The complex version thus



(a) beginning



(b) rest

Figure A.13: Derivation for an ungrammatical sentence.

captures more of the internal structure of the object relative clause, which in turn allows for incremental derivations. Incrementality can thus be obtained to a certain extent by creating deeper categories, such as the new relative pronoun, or by type-raising (which creates other forms of deep trees in that a noun phrase becomes a structures rooted in a sentence that lacks a noun phrase).

Appendix B

Traces in PLTAG

As with prediction trees, elementary trees for traces can be integrated by the parser without any evidence from the input string, and therefore can potentially slow parsing down a lot. We therefore decided to ”bind” all traces either to the tree that subcategorises them, or to the filler, depending on the relationship between filler and trace. We here discuss two different attachment strategies:

1. The trace is attached into the tree that contains the substitution node for it if

    - the substitution nodes for filler and trace are in the same elementary tree (like in passives, or simple extractions like “..., said X”- constructions).
    - the tree that the trace substitutes into adjoins into the tree that has a substitution node for the filler (such as raising or control verbs).

2. The trace and filler make up a multi-component lexicon entry (MCTAG, Weir, 1988), meaning that two elementary trees always have to be both attached into a tree, not just one of them, if

    - both the trace and the filler adjoin into the same tree (for example for extrapolate modifiers).
    - the trace-tree substitutes into a tree that in turn substitutes into the filler tree (as in relative clauses).

Cases we don’t currently treat:

- if the filler and trace substitute into different trees (can be dealt with using the non-tree-local version of MCTAG, but that makes MCTAG more expressive)

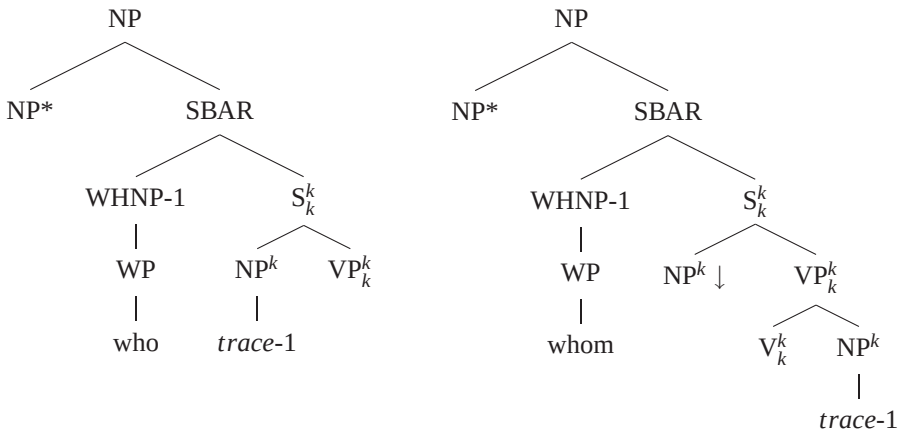
- parasitic gaps

All these cases are given with an example and discussed in more detail below.

B.1 Relative Clauses

Our current treatment of relative clauses does not encode whether the relative pronoun is a subject or object pronoun. This only becomes encoded later on with the verb entry, which will contain a trace in the subject or object position.

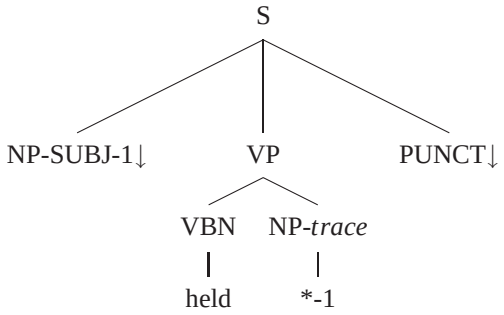
An interesting alternative might be to observe the full path between the trace and the filler. The tree for a relative pronoun would then look as follows:



We will require further experiments to establish whether the trace should is predicted as early as the relative pronoun is encountered or not. For the time being, the trace is attached to the main verb and not an MCTAG entry with the relative pronoun.

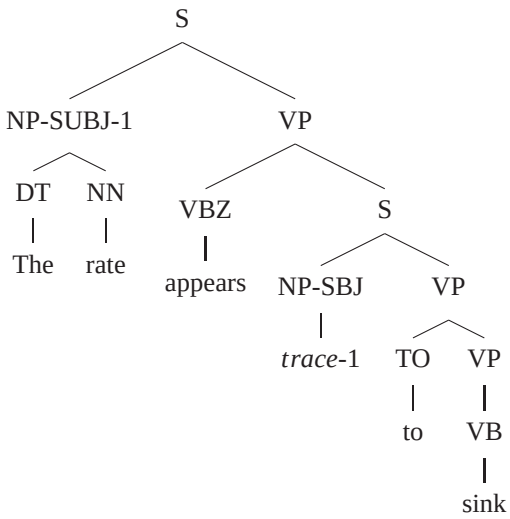
B.2 Traces for Passive Constructions

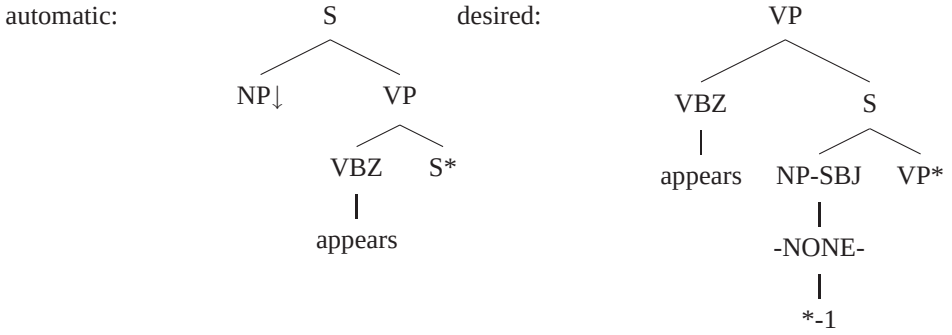
Following the above guidelines, a passive verb's lexicon entry attaches the trace at object position in the verb tree because both filler and trace are substituted into this same structure. This is attractive because it codes the filler-trace relationship in passives locally (since they are already co-indexed in the lexicon entry).



B.3 Raising and Control

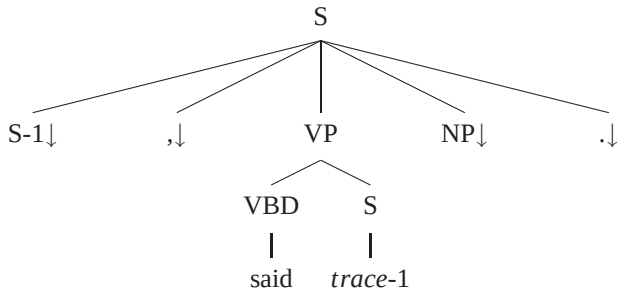
Raising constructions are slightly different, because the trace is part of the recursively adjoined tree and refers to the subject of that phrase. We'll have to decide on how to encode where to find the filler for the trace in such cases (although this may mainly be a problem of a semantic interpretation component) – the same holds for object control constructions. The trace is here only predicted once the head verb of the raising or object control construction has been encountered. Raising verbs are however still problematic because they don't have a special tag that distinguishes them from regular verbs. The automatic head finding rules based on Magerman's head percolation table determine raising verbs to be the head of the sentence and therefore assign them an incorrect structure (an auxiliary tree with the S node as a recursive structure, instead of the VP node for the recursive structure).





B.4 Extraction

Traces that occur for topicalized direct speech like in " '...', said *trace* Peter." are easy to treat because both filler and trace occur locally in the same tree.

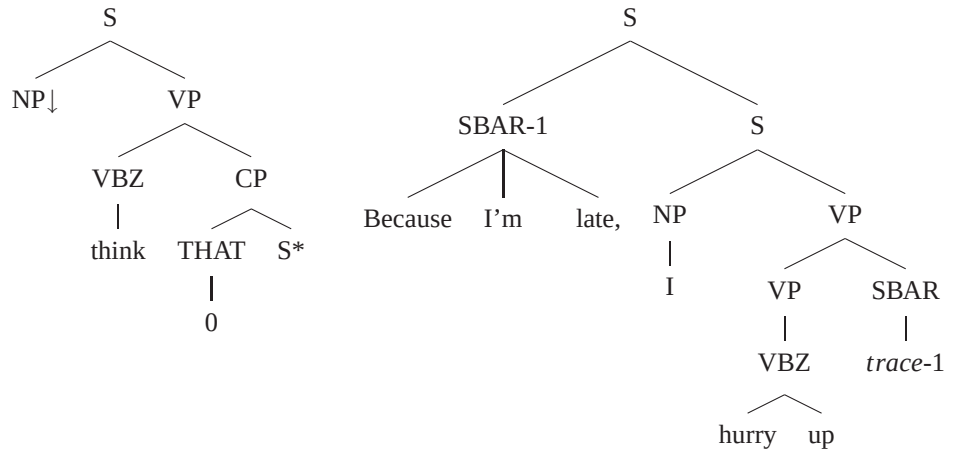


Other kinds of extraction are handled similarly, consider for example the following sentence *Because I'm late, Peter thinks 0 I hurry up,* which contains an adjunct that originated in a lower clause, and an empty element for the relative pronoun marked by *0*. However, this does not lead to a problem, because the *I think*-tree is an auxiliary tree that is adjoined in later. In terms of the lexicon however, this analysis entails that each verb must not only have an different elementary tree for each of its arguments that could be topicalized, but also for modifiers. The alternative to these multiple trees is to use multi-component TAG (MCTAG). The SBAR sentence and the trace under the VP are then simultaneously integrated at the VP. Since we require full connectivity in PLTAG, we would however have to add a prediction tree for the verb structure right away for connectivity reasons, and the effect would be very similar, perhaps attaching the trace even earlier if we decided to use lazy prediction as a parsing strategy.

If deciding against MCTAG, we run into the question of prediction grain size again: with the trace integrated into the verb structure, the trace would only be predicted once the head of the phrase (i.e. the verb, like *hurry up* in the below example tree structure) is encountered. This does not seem very plausible, in particular for the modifier example. MCTAG would on the other hand predict the trace and the structure

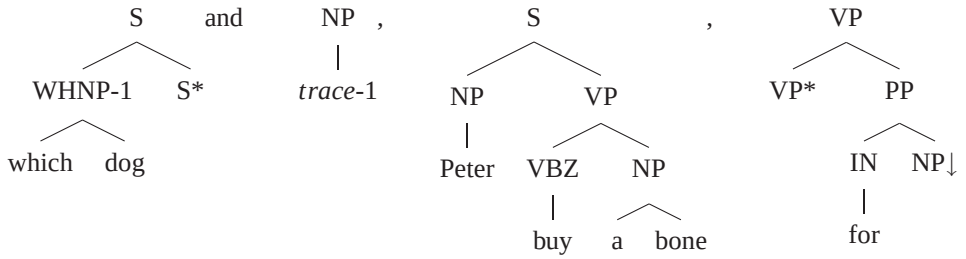
in between once the filler clause “Because...” is encountered. One important aspect to take care of when using MCTAG however is that both components have to be attached to the same elementary tree; otherwise, the formalism becomes too powerful and will over-generate. MCTAG can also explain extrapolations to the right, for example: “The man is really tall, who is wearing that hat.”

However, then we also have to learn the associated lexicon entries from the Penn Treebank, such that the filler and trace elementary trees are stored together. This phenomenon is not very common and occurs in a bit less than 1% of sentences.



B.5 Long-distance extractions

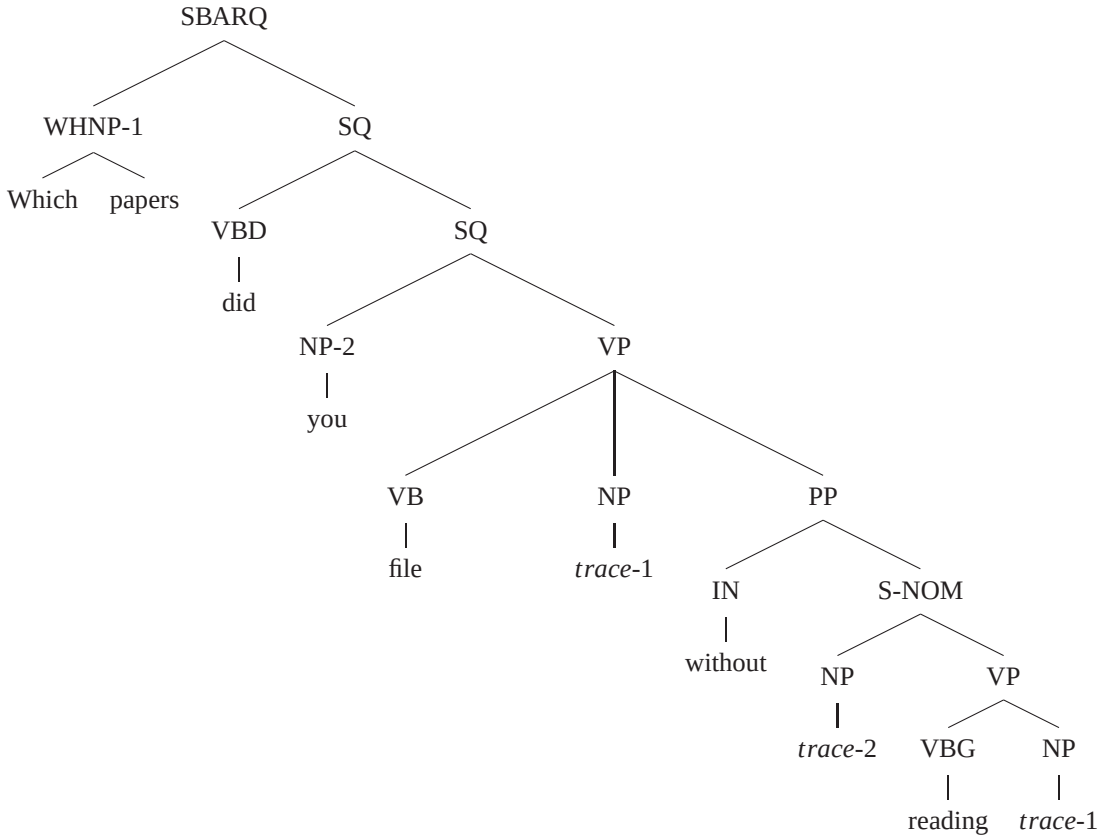
Similarly, we can use MCTAG for more deeply embedded constructions like “Which dog did Peter buy a bone for $_{trace}$?”



B.6 Parasitic gaps

A parasitic gap is a construction where a verb’s argument is dropped (i.e. replaced by a trace, the second *trace*-1 in the below example) under the condition that the co-referential argument has been fronted, also leaving a trace behind (the first *trace*-1 in the below example). The difficulty about them is that the rightmost trace is contingent on the WH-phrase having been fronted beforehand, and that this is difficult to capture

in a lexicon entry. (A sentence like “Peter files papers without reading.”) is not grammatically correct. I currently do not have a good treatment for traces due to parasitic gaps.



Bibliography

Ajdukiewicz, K. (1935). Die syntaktische Konnexität. *Studia philosophica*, 1(1):27.

Alexopoulou, T. and Keller, F. (2007). Locality, cyclicity and resumption: At the interface between the grammar and the human sentence processor. *Language*, 83(1):110–160.

Altmann, G. and Kamide, Y. (2007). The real-time mediation of visual attention by language and world knowledge: Linking anticipatory (and other) eye movements to linguistic processing. *Journal of Memory and Language*, 57(4):502–518.

Altmann, G. and Steedman, M. (1988). Interaction with context during human sentence processing. *Cognition*, 30(3):191–238.

Aoshima, S., Phillips, C., and Weinberg, A. (2004). Processing filler-gap dependencies in a head-final language. *Journal of Memory and Language*, 51:23–54.

Aoshima, S., Yoshida, M., and Phillips, C. (2007). Incremental processing of coreference and binding in japanese. *Syntax*, page 49pp.

Arai, M., Keller, F., and Demberg, V. (2008). Evidence for the prediction of modifiers during sentence processing. In *AMLaP Architectures and Mechanisms for Language Processing*, Cambridge, UK.

Baayen, R. (2008). *Analyzing linguistic data: A practical introduction to statistics using R*. Cambridge Univ Pr.

Baayen, R. H., Davidson, D. J., and Bates, D. M. (2008). Mixed-effects modeling with crossed random effects for subjects and items. *Journal of Memory and Language*, to appear.

Baldridge, J. and Kruijff, G.-J. M. (2002). Coupling ccg and hybrid logic dependency semantics. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 319–326.

Baldridge, J. M. (2002). *Lexically Specified Derivation Control in Combinatory Categorial Grammar*. PhD thesis, University of Edinburgh.

Bar-Hillel, Y. (1953). A quasi-arithmetical notation for syntactic description. *Language*, 29(1):47–58.

Bartek, B., Smith, M., Lewis, R. L., and Vasishth, S. (2007). Linking working memory processes with self-paced reading and eyetracking measures: How lexical processing load and spr exaggerate locality effects. In *Proceedings of the CUNY sentence processing conference*, La Jolla, CA.

Bates, D. M. and Sarkar, D. (2007). lme4: Linear mixed effects models using s4 classes.

Bever, T. G. (1970). The cognitive basis for linguistic structures. In Hayes, J., editor, *Cognition and the development of language*, pages 279–362. Wiley, New York.

Bicknell, K., Demberg, V., and Levy, R. (2008). Local coherences in the wild: An eye-tracking corpus study. In *CUNY Conference on Human Sentence Processing*, Chapel Hill, North Carolina.

Bicknell, K. and Levy, R. (2010). A rational model of eye movement control in reading. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics ACL. Uppsala, Sweden: Association for Computational Linguistics*.

Bikel, D. (2004). *On the parameter space of generative lexicalized statistical parsing models*. PhD thesis, University of Pennsylvania.

Bod, R., Sha, R., and Sima'an, K. (2003). *Data Oriented Parsing*. CSLI Studies in Computational Linguistics.

Boston, M. F., Hale, J. T., Kliegl, R., and Vasishth, S. (2008). Surprising parser actions and reading difficulty. In *Proceedings of ACL-08: HLT Short Papers*, pages 5–8.

Brants, T. (2000). TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, Seattle, WA.

Brants, T. and Crocker, M. (2000). Probabilistic parsing and psychological plausibility. In *Proceedings of the 18th conference on Computational linguistics*, pages 111–117, Morristown, NJ, USA. Association for Computational Linguistics.

Bresnan, J. (2001). *Lexical Functional Syntax*. Blackwell.

Bresnan, J., Kaplan, R., Peters, S., and Zaenen, A. (1982). Cross-serial dependencies in Dutch. *Linguistic Inquiry*, 13(4):613–635.

Brown, C. and Hagoort, P. (1993). The processing nature of the N400: Evidence from masked priming. *Journal of Cognitive Neuroscience*, 5(1):34–44.

Brysbaert, M. and Vitu, F. (1998). Word skipping: Implications for theories of eye movement control in reading. In Underwood, G., editor, *Eye Guidance in Reading and Scene Perception*, pages 125–147. Elsevier, Oxford.

Burnard, L. (1995). *Users Guide for the British National Corpus*. British National Corpus Consortium, Oxford University Computing Service.

Charniak, E. (1986). A neat theory of marker passing. In *Proceedings of the 5th National Conference on Artificial Intelligence (AAAI-86)*, volume 1, pages 584–588.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 132–139. Morgan Kaufmann Publishers Inc.

Chen, E., Gibson, E., and Wolf, F. (2005). Online syntactic storage costs in sentence comprehension. *Journal of Memory and Language*, 52(1):144–169.

Chen, J. (2001). *Towards efficient statistical parsing using lexicalized grammatical information*. PhD thesis, Citeseer.

Chiang, D. (2000). Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, page 463. Association for Computational Linguistics.

Chomsky, N. (1957). *Syntactic Structures*. Mouton, The Hague.

Clark, S. and Curran, J. R. (2004). Parsing the wsj using ccg and log-linear models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 104 – 111.

Clark, S. and Curran, J. R. (2007). Formalism-independent parser evaluation with ccg and depbank. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic.

Clarkson, P. R. and Rosenfeld, R. (1997). Statistical language modeling using the CMU-Cambridge toolkit. In Kokkinakis, G., Fakotakis, N., and Dermatas, E., editors, *Proceedings of the 5th European Conference on Speech Communication and Technology*, pages 2707–2710, Rhodes. International Speech Communication Association.

Cohn, T., Goldwater, S., and Blunsom, P. (2009). Inducing compact but accurate tree-substitution grammars. In *Proceedings of the Conference of the Northamerican Chapter of the Association for Computational Linguistics (NAACL)*, Boulder, Colorado.

Collins, M. (1999). *Head-driven statistical models for natural language parsing*. PhD thesis, University of Pennsylvania, Philadelphia.

Collins, M. and Roark, B. (2004). Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 111–118, Barcelona, Spain.

Coolican, H. (2004). *Research Methods and Statistics in Psychology*. Hodder Arnold, 4th edition edition.

Cristea, D. and Webber, B. (1997). Expectations in incremental discourse processing. In *Proceedings of the 8th. Conference of the European Chapter of the Association for Computational Linguistics (ACL-EACL97)*, pages 88–95, Madrid, Spain. Association for Computational Linguistics.

Crocker, M. W. and Brants, T. (2000). Wide-coverage probabilistic sentence processing. *Journal of Psycholinguistic Research*, 29:647 – 669.

Curry, H. and Feys, R. (1958). Combinatory Logic, volume I of Studies in Logic and the Foundations of Mathematics.

Delong, K. A., Urbach, T. P., and Kutas, M. (2005). Probabilistic word pre-activation during language comprehension inferred from electrical brain activity. *Nature Neuroscience*, 8(8).

Demberg, V. and Keller, F. (2007). Eye-tracking evidence for integration cost effects in corpus data. In *Proceedings of the 29th Annual Conference of the Cognitive Science Society*, pages 947–952, Nashville. Cognitive Science Society.

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

Demberg, V. and 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)*, Tübingen, Germany.

Demberg, V. and Keller, F. (2009). A computational model of prediction in human parsing: Unifying locality and surprisal effects. In *Proceedings of the 29th meeting of the Cognitive Science Society (CogSci-09)*, Amsterdam, The Netherlands.

Demberg, V. and Moore, J. (2006). Information presentation in spoken dialogue systems. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 65–72.

Dubey, A. (2004). *Statistical Parsing for German: Modeling syntactic properties and annotation differences*. PhD thesis, Saarland University.

Dubey, A., Keller, F., and Sturt, P. (2006). Integrating syntactic priming into an incremental probabilistic parser, with an application to psycholinguistic modeling. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 417–424. Association for Computational Linguistics.

Eady, J. and Fodor, J. (1981). Is center embedding a source of processing difficulty. In *Linguistics Society of America Annual Meeting*.

Engbert, R., Nuthmann, A., Richter, E. M., and l, R. K. (2005). Swift: A dynamical model of saccade generation during reading. *Psychological Review No. 4*, pages 777–813.

Federmeier, K. D. (2007). Thinking ahead: The role and roots of prediction in language comprehension. *Psychophysiology*, 44:491–505.

Ferrara Boston, M., Hale, J., Kliegl, R., Patil, U., and Vasishth, S. (2008). Parsing costs as predictors of reading difficulty: An evaluation using the Potsdam Sentence Corpus. *Journal of Eye Movement Research*, to appear.

Ferreira, F. (2005). Psycholinguistics, formal grammars, and cognitive science. *Linguistic Review*, 22(2/4):365.

Ferreira, F. and Henderson, J. (1991). Recovery from misanalyses of garden-path sentences. *Journal of Memory and Language*, 30(6):725–745.

Ferreira, F. and Henderson, J. M. (1993). Reading processes during syntactic analysis and reanalysis. *Canadian Journal of Psychology*, 47(2):247–275.

Ferreira, F. and Patson, N. (2007). The good enoughapproach to language comprehension. *Language and Linguistics Compass*, 1(1-2):71–83.

Ford, M., Bresnan, J., and R. M. Kaplan, R. M. (1982). A competence-based theory of syntactic closure. In Bresnan, I. J., editor, *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA.

Fowler, T. A. D. and Penn, G. (2010). Accurate context-free parsing with Combinatory Categorial Grammar. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics: ACL*, Uppsala, Sweden.

Frank, S. (2009). Surprisal-based comparison between a symbolic and a connectionist model of sentence processing. In *Proceedings of the 31st annual conference of the cognitive science society*, pages 1139–1144.

Frank, S. (2010). Uncertainty reduction as a measure of cognitive processing effort. In *Proceedings of the 2010 Workshop on Cognitive Modeling and Computational Linguistics*, pages 81–89, Uppsala, Sweden. Association for Computational Linguistics.

Frazier, L. (1978). *On comprehending sentences: Syntactic parsing strategies*. PhD thesis, University of Connecticut.

Frazier, L. and Clifton, C. (1998). *Reanalysis in Sentence Processing*, pages 143–176. Kluwer Academic Publishers, Netherlands.

Frazier, L. and Clifton, C. J. (1996). *Construal*. MIT Press, Cambridge, MA.

Frazier, L. and Fodor, J. D. (1978). The sausage machine: A new two-stage model of the parser. *Cognition*, 6:291–325.

Frazier, L., Munn, A., and Clifton, C. (2000). Processing coordinate structure. *Journal of Psycholinguistic Research*, 29:343–368.

Friederici, A., Steinhauer, K., Mecklinger, A., and Meyer, M. (1998). Working memory constraints on syntactic ambiguity resolution as revealed by electrical brain responses. *Biological Psychology*, 47(3):193–221.

Frisson, S., Rayner, K., and Pickering, M. J. (2005). Effects of contextual predictability and transitional probability on eye movements during reading. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 31(5):862–877.

Garnsey, S. M., Pearlmutter, N. J., Myers, E. M., and Lotocky, M. A. (1997). The contributions of verb bias and plausibility to the comprehension of temporarily ambiguous sentences. *Journal of Memory and Language*, 37(1):58–93.

Gennari, S. and MacDonald, M. (2008). Semantic indeterminacy in object relative clauses. *Journal of memory and language*, 58(2):161–187.

Gibson, E. (1991). *A Computational Theory of Human Linguistic Processing: Memory Limitations and Processing Breakdown*. PhD thesis, Carnegie Mellon University, Pittsburgh.

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

Gibson, E. (2000). Dependency locality theory: A distance-dased theory of linguistic complexity. In Marantz, A., Miyashita, Y., and O'Neil, W., editors, *Image, Language, Brain: Papers from the First Mind Articulation Project Symposium*, pages 95–126. MIT Press, Cambridge, MA.

Gibson, E. (2006). The interaction of top-down and bottom-up statistics in the resolution of syntactic category ambiguity. *Journal of Memory and Language*, 54:363–388.

Gibson, E. and Pearlmutter, N. (2000). Distinguishing serial and parallel parsing. *Journal of Psycholinguistic Research*, 29(2):231–240.

Gordon, P., Hendrick, R., and Johnson, M. (2001). Memory interference during language processing. *Learning, Memory*, 27(6):1411–1423.

Gordon, P., Hendrick, R., and Johnson, M. (2004). Effects of noun phrase type on sentence complexity. *Journal of Memory and Language*, 51(1):97–114.

Green, M. and Mitchell, D. (2006). Absence of real evidence against competition during syntactic ambiguity resolution. *Journal of Memory and Language*, 55(1):1–17.

Gundel, J., Hedberg, H., and Zacharski (1993). Referring expressions in discourse. *Language*, 69:274–307.

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*, volume 2, pages 159–166, Pittsburgh, PA. Association for Computational Linguistics.

Hale, J. (2003). The information conveyed by words in sentences. *Journal of Psycholinguistic Research*, 32(2):101–123.

Hale, J. (2006). Uncertainty about the rest of the sentence. *Cognitive Science: A Multidisciplinary Journal*, 30(4):643–672.

Hausser, R. R. (1986). *Newcat: Parsing Natural Language Using Left-associative Grammar*. (Lecture Notes in Computer Science) Berlin, New York: Springer.

Hobbs, J. R. and Bear, J. (1990). Two principles of parse preference. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING-90)*, pages 162–167, Helsinki.

Hockenmaier, J. and Steedman, M. (2002). Generative models for statistical parsing with combinatory categorial grammar. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 335–342.

Hockenmaier, J. and Steedman, M. (2007). Ccgbank: a corpus of ccg derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, 33(3):355–396.

Hoffman, B. (1995). *The Computational Analysis of the Syntax and Interpretation of Free Word Order in Turkish.* PhD thesis, University of Pennsylvania.

Honnibal, M. and Curran, J. R. (2007). Creating a systemic functional grammar corpus from the penn treebank. In *Proceedings of the ACL 2007 Workshop on Deep Linguistic Processing (DLP)*, pages 89 – 96.

Honnibal, M., Curran, J. R., and Bos, J. (2010). Rebanking ccgbank for improved np interpretation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 207–215, Uppsala, Sweden. Association for Computational Linguistics.

Jackendoff, R. (2003). *Foundations of language: Brain, meaning, grammar, evolution*. Oxford University Press, USA.

Jacqmin-Gadda, H., Sibillot, S., Proust, C., Molina, J., and Thiebaut, R. (2007). Robustness of the linear mixed model to misspecified error distribution. *Computational Statistics and Data Analysis*, 51(10):5142–5154.

Jaeger, T. F., Fedorenko, E., and Gibson, E. (2010). Anti-locality in english: Consequences for theories of sentence comprehension. *submitted*.

Johnson, C. and Fillmore, C. J. (2000). The framenet tagset for frame-semantic and syntactic coding of predicate-argument structure. In *In Proceedings ANLP-NAACL 2000*, Seattle, WA.

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

Joshi, A. and Schabes, Y. (1997). Tree-adjoining grammars. *Handbook of Formal Languages, Beyond Words*, 3:69–123.

Juhasz, B. J., Starr, M. S., Inhoff, A. W., and Placke, L. (2003). The effects of morphology on the processing of compound words: Evidence from naming, lexical decisions and eye fixations. *British Journal of Psychology*, 94(2):223–244.

Jurafsky, D. (1996). A probabilistic model of lexical and syntactic access and disambiguation. *Cognitive Science*, 20:137 – 194.

Kaan, E., Harris, A., Gibson, E., and Holcomb, P. (2000). The P600 as an index of syntactic integration difficulty. *Language and cognitive processes*, 15(2):159–201.

Kamide, Y., Scheepers, C., Altmann, G., and Crocker, M. (2002). Integration of syntactic and semantic information in predictive processing: Anticipatory eye-movements in German sentence processing. In *The Annual CUNY Conference on Sentence Processing*, New York, NY.

Kamide, Y., Scheepers, C., and Altmann, G. T. M. (2003). Integration of syntactic and semantic information in predictive processing: Cross-linguistic evidence from german and english. *Journal of Psycholinguistic Research*, 32:37–55.

Kato, Y. and Matsubara, S. (2009). Incremental Parsing with Adjoining Operation. *IEICE Transactions on Information and Systems*, 92(12):2306–2312.

Keefe, D. and McDaniel, M. (1993). The time course and durability of predictive inferences. *Journal of Memory and Language*, 32(4):446–463.

Kempson, R., Cann, R., and Marten, L. (2005). *The Dynamics of Language*. Oxford: Elsevier.

Kempson, R., Meyer-Viol, W., and Gabbay, D. (2001). *Dynamic Syntax: The Flow of Language Understanding*. Blackwell.

Kennedy, A., Brooks, R., Flynn, L., and Prophet, C. (2003). *The mind's eye: Cognitive and applied aspects of eye movement research*, chapter The reader's spatial code. Elsevier.

Kennedy, A. and Pynte, J. (2005). Parafoveal-on-foveal effects in normal reading. *Vision Research*, 45:153–168.

Kimball, J. (1973). Seven principles of surface structure parsing in natural language. *Cognition*, 2:15–47.

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

Kipper, K., Dang, H. T., and Palmer, M. (2000). Class-based construction of a verb lexicon. In *Proceedings of the 7th Conference on Artificial Intelligence (AAAI-00)*, pages 691–696.

Kliegl, R., Nuthmann, A., and Engbert, R. (2006). Tracking the mind during reading: The influence of past, present, and future words on fixation durations. *Journal of Experimental Psychology: General*, 135:12–35.

Konieczny, L. (2000). Locality and parsing complexity. *Journal of Psycholinguistic Research*, 29(6):627–645.

Konieczny, L. and Döring, P. (2003). Anticipation of clause-final heads: Evidence from eye-tracking and SRNs. In *Proceedings of ICCS/ASCS*.

Kuhlmann, M. (2007). *Dependency Structures and Lexicalized Grammars*. PhD thesis, Saarland University, Saarbrücken, Germany.

Kuhlmann, M. and Mohl, M. (2007). Mildly context-sensitive dependency languages. In *Annual Meeting – Association For Computational Linguistics*, volume 45, page 160.

Lang, B. (1991). Towards a uniform formal framework for parsing. *Current Issues in Parsing Technology*, pages 153–172.

Levy, R. (2008). Expectation-based syntactic comprehension. *Cognition*, 106(3):1126–1177.

Lewis, R. (2000). Falsifying serial and parallel parsing models: Empirical conundrums and an overlooked paradigm. *Journal of Psycholinguistic Research*, 29(2):241–248.

Lewis, R. L. and Vasishth, S. (2005). An activation-based model of sentence processing as skilled memory retrieval. *Cognitive Science*, 29:1–45.

Lin, D. (1998). An information-theoretic definition of similarity. In Shavlik, J. W., editor, *Proceedings of the 15th International Conference on Machine Learning*, pages 296–304, Madison, WI. Morgan Kaufmann.

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

Lorch, R. F. and Myers, J. L. (1990). Regression analyses of repeated measures data in cognitive research. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 16(1):149–157.

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

Mak, W., Vonk, W., and Schriefers, H. (2002). The influence of animacy on relative clause processing. *Journal of Memory and Language*, 47(1):50–68.

Mak, W. M., Vonk, W., and Schriefers, H. (2008). Discourse structure and relative clause processing. *Memory & Cognition*, 36(1):170–181.

Marcus, M. (1980). *A theory of syntactic recognition for natural language*. MIT Press, Cambrdige, MA.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Marslen-Wilson, W. and Welsh, A. (1978). Processing interactions and lexical access during word recognition in continuous speech. *Cognitive psychology*, 10(1):29–63.

Mazzei, A. (2005). *Formal and empirical issues of applying dynamics to Tree Adjoining Grammars*. PhD thesis, Dipartimento di Informatica, Universita' di Torino.

Mazzei, A., Lombardo, V., and Sturt, P. (2007). Dynamic tag and lexical dependencies. *Research on Language and Computation, Foundations of Natural Language Grammar*, pages 309–332.

McConville, M. and Dzikovska, M. O. (2008). Evaluating complement-modifier distinctions in a semantically annotated corpus. In *Proceedings of the 6th Language Resources and Evaluation Conference (LREC)*, Marrakech.

McDonald, S. A., Carpenter, R. H. S., and Shillcock, R. C. (2005). An anatomically constrained, stochastic model of eye movement control in reading. *Psychological Review*, 112(4):814–840.

McDonald, S. A. and Shillcock, R. C. (2003a). Eye movements reveal the on-line computation of lexical probabilities during reading. *Psychological Science*, 14(6):648–652.

McDonald, S. A. and Shillcock, R. C. (2003b). Low-level predictive inference in reading: The influence of transitional probabilities on eye movements. *Vision Research*, 43:1735–1751.

McRae, K., Spivey-Knowlton, M. J., and Tanenhaus, M. K. (1998). Modeling the influence of thematic fit (and other constraints) in on-line sentence comprehension. *Journal of Memory and Language*, 38(3):283–312.

Menzel, W. (2009). Towards Radically Incremental Parsing of Natural Language. *Recent Advances in Natural Language Processing V: Selected Papers from Ranlp 2007*, page 41.

Mitchell, D. C. (1987). *Attention and performance XII: The psychology of reading*, chapter Lexical guidance in human parsing: Locus and processing characteristics, pages 601–618. Hillsdale, NJ: Erlbaum.

Mitchell, J., Lapata, M., Demberg, V., and Keller, F. (2010). Syntactic and semantic factors in processing difficulty: An integrated measure. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 196–206, Uppsala, Sweden. Association for Computational Linguistics.

Morgan, E., Keller, F., and Steedman, M. (2010). A Bottom-Up Parsing Model of Local Coherence Effects. In *Proceedings of the 32nd Annual Meeting of the Cognitive Science Society*, Portland, OR.

Narayanan, S. and Jurafsky, D. (2002). A Bayesian model predicts human parse preference and reading time in sentence processing. In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14*, pages 59–65, Cambridge, MA. MIT Press.

Nivre, J. (2004). Incrementality in deterministic dependency parsing. In *Proceedings of the ACL Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona.

Padó, U. (2007). *The Integration of Syntax and Semantic Plausibility in a Wide-Coverage Model of Human Sentence Processing*. PhD thesis, Saarland University.

Palmer, M., Gildea, D., and Kingsbury, P. (2003). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Phillips, C. (2003). Linear order and constituency. *Linguistic Inquiry*, 34(1):45pp.

Pickering, M. J. and Garrod, S. (2007). Do people use language production to make predictions during comprehension? *Trends in Cognitive Sciences*, 11(3):105–110.

Pickering, M. J., Traxler, M. J., and Crocker, M. W. (2000). Ambiguity resolution in sentence processing: Evidence against frequency-based accounts. *Journal of Memory and Language*, 43(3):447–475.

Pinheiro, J. C. and Bates, D. M. (2000). *Mixed-Effects Models in S and S-PLUS*. Springer, New York.

Pollard, C. and Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago.

Potts, G., Keenan, J., and Golding, J. (1988). Assessing the occurrence of elaborative inferences: Lexical decision versus naming. *Journal of Memory and Language*, 27(4):399–415.

Purver, M. and Kempson, R. (2004). Incremental parsing, or incremental grammar? In Frank Keller, Stephen Clark, M. C. and Steedman, M., editors, *Proceedings of the ACL Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 74–81, Barcelona, Spain.

R Development Core Team (2007). *R: A language and environment for statistical computing*. ISBN: 3-900051-07-0. R Foundation for Statstical Computing, Vienna, Austria.

Ratcliff, R. (1979). Group reaction time distributions and an analysis of distribution statistics. *Psychological Bulletin*, 86(3):446–461.

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, 124(3):372–422.

Rayner, K. and Bertera, J. H. (1979). Reading without a fovea. *Science*, 206(4417):468–469.

Reali, F. and Christiansen, M. H. (2007). Processing of relative clauses is made easier by frequency of occurrence. *Journal of Memory and Language*, 53:1–23.

Reichle, E., Pollatsek, A., and Rayner, K. (2006). EZ Reader: A cognitive-control, serial-attention model of eye-movement behavior during reading. *Cognitive Systems Research*, 7(1):4–22.

Reichle, E. D., Warren, T., and McConnell, K. (2009). Using E-Z Reader to model the effects of higher level language processing on eye movements during reading. *Psychonomic Bulletin & Review*, 16(1):1–21.

Resnik, P. (1992). Left-corner parsing and psychological plausibility. In *In The Proceedings of the fifteenth International Conference on Computational Linguistics, COLING-92*, pages 191–197.

Richter, T. (2006). What is wrong with ANOVA and multiple regression? Analyzing sentence reading times with hierarchical linear models. *Discourse Processes*, 41:221–250.

Roark, B. (2001a). Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27(2):249–276.

Roark, B. (2001b). *Robust probabilistic predictive syntactic processing: motivations, models, and applications*. Brown University Providence, RI, USA.

Roark, B., Bachrach, A., Cardenas, C., and Pallier, C. (2009). Deriving lexical and syntactic expectation-based measures for psycholinguistic modeling via incremental top-down parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 324–333, Singapore. Association for Computational Linguistics.

Roland, D. (2008). Relative clauses and surprisal. In *Poster presented at the 14th Annual Conference on Architectures and Mechanisms for Language Processing*.

Roland, D. (2009). Relative clauses remodeled: The problem with mixed-effect models. In *Poster presented at the CUNY 2009 Conference on Human Sentence Processing*.

Roland, D., OMeara, C., Yun, H., and Mauner, G. (2007). Processing object relative clauses: Discourse or frequency. In *Poster presented at the CUNY sentence processing conference. La Jolla, CA*.

Rozenberg, G. and Salomaa, A. (1997). *Handbook of Formal Languages: Beyond Words*. Springer Verlag.

Sag, I. A. and Fodor, J. D. (1994). Extraction without traces. In *Proceedings of the Thirteenth West Coast Conference on Formal Linguistics*, pages 365–384, Stanford, CA. CSLI Publications.

Sampson, G. (1995). *English for the Computer: The SUSANNE Corpus and Analytic Scheme*. Clarendon Press, Oxford.

Sarkar, A. (2001). Applying co-training methods to statistical parsing. In *Proceedings of the 2nd Meeting of the North American Association for Computational Linguistics: NAACL*, pages 175–182, Pittsburgh, PA.

Schabes, Y. and Waters, R. C. (1995). Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Fuzzy Sets Syst.*, 76(3):309–317.

Schubert, L. K. (1984). On parsing preferences. In *Proceedings of the Tenth International Conference on Computational Linguistics*, pages 247–250.

Schuler, W., Miller, T., AbdelRahman, S., and Schwartz, L. (2008). Toward a psycholinguistically-motivated model of language processing. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 785–792. Association for Computational Linguistics.

Shen, L. and Joshi, A. K. (2005). Incremental ltag parsing. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 811–818.

Shieber, S. (1985). Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8(3):333–343.

Shieber, S. M. and Johnson, M. (1993). Variations on incremental interpretation. *Journal of Psycholinguistic Research*, 22(2):287–318.

Singer, M. and Ferreira, F. (1983). Inferring consequences in story comprehension. *Journal of Verbal Learning and Verbal Behavior*, 22(4):437–448.

Singer, M., Graesser, A., and Trabasso, T. (1994). Minimal or global inference during reading. *Journal of Memory and Language*, 33:421–441.

Staub, A. (2007). The parser doesn't ignore intransitivity, after all. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 33:550–569.

Staub, A. (2010). Eye movements and processing difficulty in object relative clauses. *Cognition*, 116:71–86.

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

Steedman, M. (1996). *Surface structure and interpretation*. MIT press.

Steedman, M. (2000). *The syntactic process*. The MIT press.

Steedman, M. and Altmann, G. (1989). Ambiguity in context: A reply. *Language and Cognitive Processes*, 4:105–122.

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

Sturt, P. and Yoshida, M. (2008). The speed of relative clause attachment. In *Proceedings of the 14th Annual Conference on Architectures and Mechanisms for Language Processing*, Cambridge, UK.

Swets, B., Desmet, T., Clifton, C., and Ferreira, F. (2008). Underspecification of syntactic ambiguities: Evidence from self-paced reading. *Memory and Cognition*, 36:201–216.

Swinney, D. A. and Cutler, A. (1979). The access and processing of idiomatic expressions. *Journal of Verbal Learning and Verbal Behavior*, pages 523–534.

Tabor, W., Galantuccia, B., and Richardson, D. (2004). Effects of merely local syntactic coherence on sentence processing. *Journal of Memory and Language*, 50(4):355–370.

Tabor, W. and Hutchins, S. (2004). Evidence for self-organized sentence processing: Digging in effects. *Learning, Memory*, 30(2):431–450.

Tabor, W., Juliano, C., and Tanenhaus, M. (1997). Parsing in a dynamical system: An attractor-based account of the interaction of lexical and structural constraints in sentence processing. *Language and Cognitive Processes*, 12(2):211–271.

Tabor, W. and Tanenhaus, M. (2001). Dynamical systems for sentence processing. *Connectionist psycholinguistics*, pages 177–211.

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

Tesnière, L. (1959). *Elements de syntaxe structurale*. Editions Klincksieck.

The XTAG Research Group (2001). A lexicalized tree adjoining grammar for english. Technical report, Institute for Research in Cognitive Science, University of Pennsylvania.

Thompson, H. S., Dixon, M., and Lamping, J. (1991). Compose-reduce parsing. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 87–97, Berkeley, California.

Traxler, M., Morris, R., and Seely, R. (2002). Processing Subject and Object Relative Clauses: Evidence from Eye Movements. *Journal of Memory and Language*, 47(1):69–90.

Traxler, M., Williams, R., Blozis, S., and Morris, R. (2005). Working memory, animacy, and verb class in the processing of relative clauses. *Journal of Memory and Language*, 53(2):204–224.

Traxler, M. J. (2007). Working memory contributions to relative clause attachment processing: A hierarchical linear modeling analysis. *Memory and Cognition*.

Traxler, M. J., Pickering, M. J., and Clifton, C. (1998). Adjunct attachment is not a form of lexical ambiguity resolution. *Journal of Memory and Language*, 39:558–592.

Trueswell, J. C., Tanenhaus, M. K., and Kello, C. (1993). Verb-specific constraints in sentence processing: Separating effects of lexical preference from garden-paths. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 19(3):528–553.

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

van Berkum, J., Hagoort, P., and Brown, C. (1999a). Semantic integration in sentences and discourse: Evidence from the N400. *Journal of Cognitive Neuroscience*, 11(6):657–671.

van Berkum, J., Koornneef, A., Otten, M., and Nieuwland, M. (2007). Establishing reference in language comprehension: An electrophysiological perspective. *Brain Research*, 1146:158–171.

van Berkum, J. J., Brown, C., Zwitserlood, P., V. Kooijman, V., and Hagoort, P. (2005). Anticipating upcoming words in discourse: Evidence from erps and reading times. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 31:443–467.

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

van Gompel, R. P., Pickering, M. J., Pearson, J., and Liversedge, S. P. (2005). Evidence against competition during syntactic ambiguity resolution. *Journal of Memory and Language*, 52:284–307.

Vasishth, S. and Lewis, R. L. (2006). Argument-head distance and processing complexity: Explaining both locality and antilocality effects. *Language*, 82(4):767–794.

Vijay-Shankar, K. and Joshi, A. K. (1986). Some computational properties of tree adjoining grammars. In *HLT '86: Proceedings of the workshop on Strategic computing natural language*, pages 212–223, Morristown, NJ, USA. Association for Computational Linguistics.

Vijay-Shanker, K. and Joshi, A. K. (1988). Feature structures based tree adjoining grammars. In *Proceedings of the 12th conference on Computational linguistics*, pages 714–719, Morristown, NJ, USA. Association for Computational Linguistics.

Vijay-Shanker, K. and Weir, D. J. (1994). The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:27–511.

Vitu, F., McConkie, G., Kerr, P., and O'Regan, J. (2001). Fixation location effects on fixation durations during reading: An inverted optimal viewing position effect. *Vision Research*, 41(25-26):3513–3533.

Warren, T. and Gibson, E. (2002). The influence of referential processing on sentence complexity. *Cognition*, 85(1):79–112.

Weir, D. (1988). *Characterizing Mildly Context-Sensitive Grammar Formalisms*. PhD thesis, Department of Computer and In- formation Science, University of Pennsylvania.

Wells, J., Christiansen, M., Race, D., Acheson, D., and MacDonald, M. (2009). Experience and sentence processing: Statistical learning and relative clause comprehension. *Cognitive psychology*, 58(2):250–271.

Witten, I. H. and Bell, T. C. (1991). The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transaction on Information Theory*, 37(4):1085–1094.

Wu, S., Bachrach, A., Cardenas, C., and Schuler, W. (2010). Complexity metrics in an incremental right-corner parser. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1189–1198, Uppsala, Sweden. Association for Computational Linguistics.

Xia, F., Palmer, M., and 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*, pages 53–62.

Yoshida, M., Walsh-Dickey, M., and Sturt, P. (2009). Sluicing and syntactic prediction. *In Preparation*.