A U-DOP approach to modeling language acquisition

MSc Thesis (*Afstudeerscriptie*)

written by

Margaux Smets
(born September 17th, 1987 in Tienen, Belgium)

under the supervision of **Prof Dr Rens Bod** and **Federico Sangati, MSc**, and submitted to the Board of Examiners in partial fulfillment of the requirements for the degree of

MSc in Logic

at the *Universiteit van Amsterdam.*

Date of the public defense:	**Members of the Thesis Committee:**
August 31, 2010	Prof Dr Rens Bod
	Prof Dr Frank Veltman
	Dr Jelle Zuidema
	Federico Sangati, MSc

Institute for Logic, Language and Computation

Abstract

In linguistics, there is a debate between empiricists and nativists: the former believe that language is acquired from experience, the latter that there is an innate component for language. The main arguments adduced by nativists are Arguments from Poverty of Stimulus. It is claimed that children acquire certain phenomena, which they cannot learn on the basis of experience alone —and therefore, there has to be some innate component for language. In this thesis, we show that at least for certain phenomena that are often used in such arguments, it *is* possible to explain how children acquire them on the basis of experience alone, viz. with an Unsupervised Data-Oriented Parsing (U-DOP) approach to language.

In the first part of the thesis, we develop concrete implementations of U-DOP, and contribute to the field of unsupervised parsing with two innovations. First, we develop an algorithm that performs syntactic category labeling and parsing simultaneously, and second, we devise a new methodology for unsupervised parsing, which can in principle be applied to any unsupervised parsing algorithm, and which produces the best results reported on the ATIS-corpus so far, with a promising outlook for even better results.

In the second part of the thesis, we then use these concrete implementations to show how the acquisition of certain phenomena can be explained in an empirical way. We look in detail at wh-questions, and then show that the U-DOP approach is more general than the nativist account by looking at other phenomena.

Acknowledgments

In the first place, I wish to thank Rens Bod, my main supervisor, for introducing me to the framework of Data-Oriented Parsing and making me enthusiastic about it, for his guidance and inspiration, and for motivating me to do original research, rather than merely studying what others have done.

I also want to thank Federico Sangati, my other supervisor, for the inspiring discussions and for helping me with all the technical problems I encountered.

Speaking of technical problems, I want to thank the entire Vici-group for letting me use the Vici-machine, and putting up with me all the times my processes managed to kill theirs. Also, I want to thank the group for the interesting coffee meetings, which really made me feel part of a research community.

I want to thank Frank Veltman and Jelle Zuidema for accepting to be on the thesis committee and to go through this entire thesis.

Two years ago, I couldn't imagine I would be where I am now, and that is all thanks to the fantastic Master in Logic program. During this two-year master, I have learned more than I thought was possible, and I was surprised to find myself pushing the limits of what I thought I could do every day. Therefore, I want to thank the entire ILLC for offering this excellent program and providing a stimulating research environment. In particular, I want to thank Tanja Kassenaar, who helped me with all administrative difficulties I encountered, and Inés Crespo, for being a great mentor.

Finally, I want to give a special thanks to Lorenz Demey, for stimulating me to come to the ILLC and choose for an academic life in the first place, for taking an effort to understand what I'm studying, and then helping me with all kinds of problems I encountered, for staying with me in good times and in bad, and for so many other things, but most importantly, for always being there for me.

Contents

Chapter 1

Introduction

This introduction consists of three sections. First, we will sketch the general background within which this thesis is to be situated. Next, we will make explicit what the main aim of this thesis is. Finally, we will give an overview of the structure of the thesis.

1.1 Background

Current linguistics can roughly be divided into two streams: nativist and empiricist approaches. Nativist approaches claim that humans are endowed with a special 'language faculty', encoded on the gene, which distinguishes humans from other animals. They have a *Platonic* view on language acquisition: the main *principles* and *parameters* are already innate as ideas in the child's mind; language acquisition proceeds as a Socratic dialogue, where experience merely serves to bring the ideas, already lingering in the background, to the fore. Empiricist approaches, on the other hand, try to refrain from assuming innate ideas, and instead try to show how children can learn language solely on the basis of (i) experience and (ii) general cognitive capabilities (which are not unqiue to language, but are also used in music, visual perception, reasoning, etc.). However, this does not imply the following:

> To say that 'language is not innate' is to say that there is no difference between my granddaughter, a rock and a rabbit. In other words, if you take a rock, a rabbit and my granddaughter and put them in a community where people are talking English, they'll all learn English. If people believe that, then they believe that language is not innate. If they believe that there is a difference between my granddaughter, a rabbit and a rock, then they believe that language is innate. (Chomsky, 2000, p. 50)

Whereas empiricists try not to assume any innate linguistic principles and parameters, they do however assume that certain *general cognitive capabilities*

are innate. What distinguishes Chomsky's granddaughter, a rabbit and a rock are then precisely these general cognitive capabilities. An example of such a capability is the ability to perform analogical operations, cf. the work of Dedre Gentner (e.g. Gentner (1997)). Tomasello (2003) shows how this general notion of analogy is used in language acquisition. In Bod (2009), analogy is presented as the basis of the computational model of language learning presented there.

Arguments from Poverty of Stimulus. Nativist approaches to language acquisition are often argued for using *Arguments from Poverty of Stimulus* (APS). For a detailed and precise definition, see Pullum and Scholz (2002). Typically, such arguments run as follows. It is observed (i) that children can master a certain language phenomenon (e.g. long-distance wh-questions), and (ii) that the input they receive is too poor to explain how they mastered that phenomenon (e.g. because they have never heard that kind of construction before). It is then concluded that (iii) since they cannot have learned the phenomenon on the basis of input alone, there has to be some kind of innate component which *can* explain how they learned the phenomenon (e.g. the island constraints).

The crucial step in such an argument is of course step (ii): for the argument to go through, the nativist must ensure that this step is backed up. In this thesis, we will show that at least for certain phenomena which have traditionally played a role in APS's it *is* possible to explain their acquisition on the basis of input alone. Hence, they can no longer be used in an APS, since step (ii) fails.

Nativist approaches to language. Nativism is closely linked with the tradition of generative grammar. This tradition was started in the 1950s by Noam Chomsky under the name *transformational grammar*. The core idea of this theory was that sentences have a deep structure and a surface structure, and that syntactic theory consists of transformations that transform the deep structure into the surface structure. An important document for this linguistic tradition is the PhD thesis of Ross (1967), in which the *island constraints* were first formulated.

In the 1980s, transformational grammar gradually became obsolete, and the new *Principles and Parameters theory* or *Government and Binding* became the dominant framework in the generative tradition. More recently (initiated in 1993), the paradigm has shifted again, this time to the *Minimalist Program*. The Minimalist Program assumes that the structures proposed by Principles and Parameters theory are in essence correct, but it wants to explain why the theory works the way it does.

In this thesis, we will focus on the one hand on the roots of generative grammar, more specifically on the island constraints proposed by Ross (1967), and on the other hand on more recent work, guided by the version of the minimalist framework developed in Adger (2003).

Empiricist approaches to language. There are basically two kinds of empiricist approaches to language: theoretical and computational. The difference between both is not as much ideological as it is teleological. The aim of theoretical approaches to language is to establish what the theory of language is that humans employ, or the mechanism of language acquisition that is actually going on in infants. Computational approaches, on the other hand, take a more

behavioristic stance: the ultimate goal is to *simulate* language, to let computers do whatever it is that humans do, albeit perhaps not in the same way.

The main theoretical empiricist approaches to language are Usage-Based Grammar and Construction Grammar. These will be discussed in more detail in Section 2.1 below.

In computational empiricist approaches to language, there are two streams: symbolic and non-symbolic approaches. The main example of the latter is *connectionism*. Connectionism tries to model language with neural networks. Its main characteristic is that it is not hierarchical: unlike in more 'traditional' linguistics, there is no notion of 'tree'; language consists only of sequences of strings. An immediate consequence of this is that discontinuous relations among constituents of a sentence (e.g. the relation between 'pick' and 'up' in 'Jacob picked Bella up') cannot be captured. More problematic from a modeling point of view, however, is the restriction that such models can only *recognize* sentences; they cannot *generate* sentences. So they can only say of a given sentence whether it is grammatical; they cannot generate grammatical sentences themselves. Attempts at modeling language acquisition in this framework have already been made, e.g. in MacWhinney (2004). However, up to now this kind of work has only been conducted with simple artificial examples, and not yet with full-scale natural language.

In this thesis, we will work in the symbolic framework of Data-Oriented Parsing (DOP) (cf. Section 2.2 for more details). Unlike in connectionism, the notion of 'tree' *does* play an important role here. Consequently, discontinuous relations can be captured. Moreover, this framework also allows for the *generation* of sentences (for a discussion, see Section 2.3), and can be used with actual natural language, making it more adequate for modeling language acquisition.

Originally, this framework was primarily a *computational model*, used for applications in computational linguistics. Recently, however, the framework has been evolving towards a *theory of language*, i.e. also claiming theoretical relevance. For example, Bod (2009) already showed how U-DOP can be used to model language acquisition, via computational experiments and an investigation of the Subject Auxiliary Inversion-problem (cf. Chapter 5). This thesis wants to contribute to this evolution, by investigating more problems in language acquisition, and accounting for them in an empirical way, in the (U-)DOP-framework.

Recently, Waterfall et al. (2010) have conducted related work: they also propose a computational model of language acquisition. The main difference between our approach and theirs is that they do not really address specific problems in language acquisition, but focus more on the computational side: which sentences can be generated, with which precision and recall.

1.2 Aims and methodology

In this section, we will discuss the aims of this thesis, and explain the methodology we used to reach them. Then we discuss the necessary limitations of this

research, and the materials that we used.

1.2.1 Aims

The aims of this thesis are twofold. The main goal is to provide counterevidence to Arguments from Poverty of Stimulus. Step (ii) of such APS's says that for certain phenomena it is not possible to explain how children learn them on the basis of input alone. We will show that it *is* possible to explain these phenomena on the basis of input and a few very general cognitive capabilities, viz. in the (U-)DOP framework. Note that we do not claim that the solution proposed here is *the* correct account of the acquisition of such phenomena; rather, we claim that is a *possible* account, and that hence step (ii) of many APS's is falsified.

To show that an empirical account of these phenomena is possible, we need to build concrete implementations of the Data-Oriented Parsing framework. These implementations will be unsupervised parsers, and this brings us to the next goal of this thesis: we want to contribute to the field of unsupervised parsing, by providing new implementations which obtain better scores than the current state-of-the-art.

1.2.2 Assumptions and method

To achieve the main goal, an empirical account of phenomena in language acquisition, we take over the assumptions implicit in the (Unsupervised-)Data-Oriented-Parsing framework: that language expressions are represented as hierarchical trees, and that new utterances are formed by combining fragments of trees (subtrees), in line with the general cognitive capability to perform analogical operations. Crucially, this system depends mostly on the *input* the learner receives, rather than on principles which are already innate.

To account for the phenomena that are typically used in APS's, we proceed as follows. We train the parser developed in Section 3.1 on (a portion of) the actual input a child receives (cf. Subsection 1.2.4), to model the learning process. Then we need to explain why, when faced with several alternatives, children choose the correct alternative. For example, to formulate a yes/no-question (cf. Chapter 5), children can choose between (1.1) and (1.2); we have to explain why they utter the grammatical (1.1) and not the ungrammatical (1.2).

(1.1) is the boy who is eating hungry

(1.2) * is the boy who eating is hungry

At this point, we need to comment on the notion of grammaticality, which is different in the nativist framework and in the DOP-framework. In the nativist framework, grammaticality is considered *absolute*: a sentence is either grammatical or it is not. In the DOP-framework, however, grammaticality is considered *relative*: all sentences can be generated, but some are more likely to be generated (are more grammatical) than others. So the task is not so much

to explain why (1.1) is grammatical and (1.2) is ungrammatical, but rather to explain why children seem to prefer (1.1) over (1.2).

We let our implementation provide a syntactic analysis for both alternatives, and look at the score of each derivation. The derivation with the best score is to be preferred over the other one; so the chosen alternative is the sentence with the best-scoring derivation.[1]

Crucially, we look at the *score* of each derivation, not at the *tree* it produces as output. Since we cannot know which tree would be the tree constructed by the child, we prefer to be agnostic as to what is the 'correct' tree; therefore, we consider it irrelevant whether the parser produces the same tree a linguist would propose. Rather we focus on the relative scores: it *is* possible to know which derivation should have the highest score, viz. the derivation of the alternative that is considered most grammatical.

Problematic with this methodology is that some sentences could not be parsed by the implementations due to memory issues (if the process required more than 30 GB RAM). Therefore, the account of phenomena in language acquisition is a two-pass model. In the first pass, we try the methodology outlined above, with the concrete implementation of Section 3.1. However, when this proved to be impossible due to memory issues (more than 30 GB RAM required), we backed off to a simpler model in the second pass.

In this second pass, we manually checked which were the shortest derivations (although this could also be implemented computationally). We proceed as follows. First, we check whether the entire tree occurs in the corpus; if so, then this is the shortest derivation, consisting of one step. If we cannot find the entire tree, we look for the largest fragments in the tree that can be found in the corpus, and build our derivation with those. So in this second pass, we do not look at the k-best shortest derivations and the ranking. We only look for a shortest derivation, i.e. a derivation such that it is not possible to find a derivation that is strictly shorter. Although the parser (with sufficient memory) could come up with a derivation we hadn't thought of, we ensure that at least it cannot produce an even shorter derivation.

1.2.3 Limitations

In this subsection, we discuss the limitations of the approach adopted in this thesis.

The first limitation is that we do not take semantics into account. The (U-) DOP models we work with in this thesis only deal with syntax and not with semantics. There are already models in the DOP-framework which consider semantics (e.g. Bod and Kaplan (2003)), but their implementation still needs further research.

Second, we assume as input for the parser a corpus of sentences with part-of-speech annotation. This is a common assumption in the field of unsupervised

[1] However, it is often the case that longer sentences have longer derivations and a higher ranking score; so when we compare two sentences with a different length, this might be an issue.

parsing. However, this does not imply that the system cannot be fully unsupervised: unsupervised part-of-speech taggers are available (e.g. Schütze (1995) and Biemann (2006)), so they can be used in a preprocessing step.

Third, we often run into technical limitations: some processes either take too much time, or too much memory to complete. For this reason, we have introduced the two-pass model (cf. supra) to account for phenomena in language acquisition. Ideally, we would want to account for everything using the implementations. However, it is important to note that there are no *fundamental* problems with the implementations: if the technology would be more advanced (the working memory larger), they could run without limitations.

1.2.4 Materials

We used two corpora, for the two different parts of this thesis. For the first part, building concrete implementations of U-DOP, we used the ATIS corpus (Marcus et al., 1993) to evaluate the implementations, and compare the results to previous work. This small corpus contains around 500 utterances asking for travel information. Although most sentences are fairly small, it is still a relatively difficult corpus for parsing, because the utterances are spontaneous speech.

For the second part, explaining phenomena in language acquisition, we used the child-directed speech from the Adam-part of the Childes-corpus (Brown, 1973). This is real-life data, produced by caretakers to the child; so we have direct access to the input the child receives. Remarkably, this only represents a fraction of the real amount of input a child receives (roughly two hours per fortnight), so the fact that we can already explain phenomena in language acquisition on the basis of this small fraction is further support for our approach.

In this thesis, we will use the manually annotated part-of-speech tags of each corpus. A list of part-of-speech tags for each corpus can be found in the Appendix. We preferred using the manual tags, rather than the output of an unsupervised tagger, because we wanted to avoid errors due solely to the tagger. However, using the manual tags also has its disadvantages, as will become clear in Subsection 6.3.5: it might be desirable to have a more fine-grained system.

1.3 Overview of the thesis

In this section we will provide an overview of the remainder of the thesis. The thesis is mainly divided into two parts, reflecting its two research goals.

The first part aims at developing an empiricist theory of language in the framework of Data-Oriented Parsing, which can then be used in the second part to explain language phenomena in an empirical way. First, the theoretical background for the implementations is sketched. We look at Usage-Based Grammar and Construction Grammar, fairly theoretical linguistic theories to which Data-Oriented Parsing is closely related. Then we introduce the Data-Oriented Parsing framework itself, and show how it can be extended towards

Unsupervised Data-Oriented Parsing (U-DOP).

In the next chapter we will then build concrete implementations on the basis of these theoretical ideas. We will develop four implementations, two of which are innovations to the field of unsupervised parsing. The first innovation is to do the tasks of syntactic category labeling and parsing simultaneously, in the hope that both tasks will benefit from the interaction. The second innovation is a new methodology for unsupervised parsing in general, which will give the best results reported on the ATIS-corpus so far.

In the second part we will then look at specific language phenomena that are typically used in Arguments from Poverty of Stimulus, and use the implementations of the first part to show that an empiricist account of these phenomena can be given.

An introductory chapter provides an overview of the basics of the minimalist framework, as developed in Adger (2003). These basics will be necessary to explain the minimalist account of the language phenomena in the next chapters.

The next chapter deals with the phenomenon of Subject Auxiliary Inversion (SAI). Bod (2009) has already shown how this phenomenon can be captured empirically; in this chapter, we will investigate whether our approach can also capture this phenomenon.

Then we move to the main chapter of this thesis, on the phenomenon of wh-questions. This phenomenon will be studied in depth. First, we will look at all the details of the nativist account provided by Ross (1967) and the minimalist framework developed in Adger (2003). Then we will show how the (U-)DOP-account can capture all the details of this construction in a simple and uniform way.

Next, we will look at related phenomena, and investigate whether the (U-)DOP-approach is (i) equally general as the approach developed by Ross (1967), in that it can be extended to capture other phenomena as well and (ii) even more general, in that it can also capture phenomena which fall outside the scope of Ross' theory.

Finally, we will sum up the results from this thesis (both computational and theoretical), and look at questions for further research.

Part I

Developing a U-DOP theory of language

In this part, we will develop a U-DOP theory of language. In the first chapter, we look at the theoretical side. We discuss Usage-Based Grammar and Construction Grammar, fairly theoretical linguistic framework that are closely related to the (more computational) Data-Oriented Parsing framework. Next, we look at the basic Data-Oriented Parsing model, and show how it can be extended towards Unsupervised Data-Oriented Parsing.

In the next chapter, we will then turn towards concrete implementations of the ideas given in the first chapter. First, we will implement the basic idea of shortest derivation U-DOP, as developed in the last section of the first chapter. Next, we will optimize this implementation, leading to a small loss in F1-score, but a great gain in space and time efficiency. In the third and fourth sections, we will present two real innovations to the field of unsupervised parsing. The first innovation is to do the tasks of parsing and category labeling in one go, in the hope that this will be to the benefit of both tasks. The second innovation has not so much to do with concrete parsing techniques, but rather proposes a general new methodology for unsupervised parsing. We will show how this new methodology improves the results of the parsers developed in this thesis, but in principle the methodology can be applied to and improve the results of any existing unsupervised parser.

In the next part, we will then use these concrete implementations of the ideas of U-DOP to show that certain problems of language acquisition can be solved in an empirical way.

Chapter 2

Theoretical Background

In this chapter, we develop the theory of language that will be adopted in this thesis from a theoretical point of view. First, we look at two theoretical linguistic frameworks that are closely related to the Data-Oriented Parsing framework: Usage-Based Grammar (Bybee, 2006) and Construction Grammar (Goldberg, 2003). Next, we look at the basic Data-Oriented Parsing model, which was introduced informally by Scha (1990) and then formally developed in Bod (1992) and Bod (1993). Finally, we discuss the principles of Unsupervised Data-Oriented Parsing, building on Bod (2009). In the next chapter, we will then build concrete implementations, and extend the basic model in various ways.

2.1 Usage-Based Grammar and Construction Grammar

Usage-Based Grammar and Construction Grammar are primarily theoretical frameworks; we discuss them here because they constitute a general framework, of which the Data-Oriented Parsing approach could be considered a computational implementation. First, we discuss the key components of these frameworks; in the next section, we will discuss the basics of Data-Oriented Parsing and show how it relates to Usage-Based Grammar and Construction Grammar.

The major characteristic of these theories of language is that they are *usage-based*. Earlier generative approaches to linguistics made a distinction between competence and performance, i.e. between the internal, abstract knowledge of a language, and the manifest, concrete use of a language, and focused on the former. Usage-Based Grammar and Construction Grammar, however, consider language use the key to our understanding of language: "grammar is the cognitive organization of one's experience with language" (Bybee, 2006, p. 711). So it is language use that constitutes the grammar, and not vice versa. Thus, grammar *emerges* from experience.

The representation of the grammar is another key point of Usage-Based and

Constructionist theories. Grammar is conceived as a "construct-i-con" (Goldberg, 2003, p. 219), a network of *constructions*. Constructions are form-meaning pairings of linguistic patterns with varying size and level of abstractness. So fairly concrete, fixed patterns such as idioms are constructions (e.g. (2.1)), but also more abstract, partially specified patterns, such as (2.2), and even very abstract configurations such as the ditransitive construction, as in (2.3).

(2.1) a penny for your thoughts

(2.2) jog ⟨POSS⟩ memory

(2.3) ⟨AGENT⟩ ⟨DITRANSITIVE VERB⟩ ⟨THEME⟩ ⟨GOAL⟩

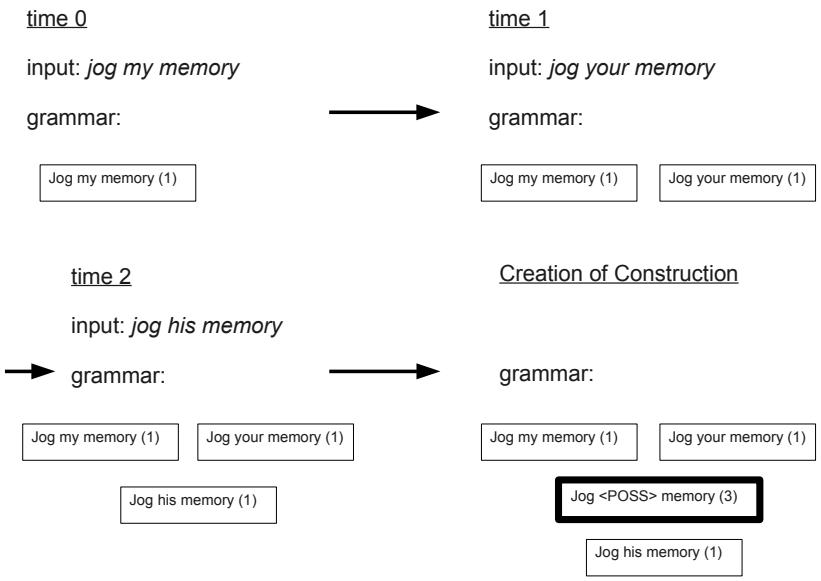
In the network, the constructions are represented as *exemplars*, complex items containing phonological, morphological, syntactic, semantic and pragmatic information. When a person encounters a language event, every token is classified and placed in the network. When a token is identical to an already existing exemplar, it is mapped onto that exemplar, altering it in the process. When it is not identical, but similar to an existing exemplar, it is placed near it, creating clusters of similar exemplars. Out of these clusters, constructions can arise, cf. Figure 2.1. *Frequency* is considered very important in the Usage-Based and Constructionist theories of language, and is stored for each exemplar (see e.g. Bybee (2006)). When a new token is mapped onto an exemplar, the exemplar is *strengthened*, because its frequency is increased.

An important difference with generative approaches to language is the importance of (semi-)idiosyncratic expressions such as idioms. In generative grammar, a distinction is made between the core and the periphery of a language. All phenomena which are more or less idiosyncratic are considered to belong to the periphery, and the focus of investigation lies on the core, i.e. the 'regular' expressions. In Usage-Based Grammar and Construction Grammar, it is the *periphery* that is most important: if we can explain these 'difficult' cases, we should certainly be able to explain the 'easy' cases with the same mechanisms.

The production of a new utterance takes place through the combination of various constructions. For example, out of construction (2.1) it is not possible to create a new utterance. However, out of (2.2) a whole range of new utterances can be construed (e.g. jog Bella's/Edward's/the Pope's/the guy with the nice car his/... memory). In this way, Usage-Based and Constructionist theories can account for the infinite creativity of natural language.

With respect to language learning, these theories hold an entirely different view from traditional generative approaches. These argue that language learners have an innate component, specifically for language, a so-called 'universal grammar'. Moreover, the only role empirical input has to play is in setting the parameters that are already hard-wired. So the learning mechanism is *language-specific* and the role of input is limited. In Usage-Based Grammar and Construction Grammar, however, it is argued that "language *must* be learnable from positive input together with fairly general cognitive abilities" (Bybee, 2006, p. 222). So the learning mechanisms are *not* language-specific, and empirical input is the basis of all learning.

Figure 2.1: Emergence of a construction



To summarize, the main characteristics of the Usage-Based Grammar and Construction Grammar approaches to language are listed in Table 2.1.

2.2 Data-Oriented Parsing

In this section, we will discuss the basic philosophy of Data-Oriented Parsing (DOP). This was introduced informally first by Scha (1990); we will look at the simple DOP1/Tree-DOP models for *supervised* parsing developed in Bod (1992) and Bod (1993),[1] and the non-probabilistic model using the shortest derivation introduced by Bod (2000); in the next section, we will then discuss the principles of *unsupervised* parsing in the DOP-framework.

[1]For a recent implementation of DOP that is efficient and achieves high accuracy, see Bansal and Klein (2010).

Table 2.1: The main characteristics of Usage-Based Grammar and Construction Grammar

usage-based	emphasis on performance/language use grammar emerges from experience
basic unit	constructions represented as exemplars importance of frequency focus on periphery rather than core
production	combination of constructions
learning	general capabilities (not language-specific) from experience

Data-Oriented Parsing is essentially an *exemplar-based* approach: it assumes that "language understanding and production operates on a store of exemplars" (Bod, 2006, p. 292). Very broadly, an exemplar is a representation of a language experience. More specifically, in DOP, phrase-structure trees are chosen as representations. Significantly, these trees represent the *surface* constituent structure: there is no such thing as deep structure (unlike in generative grammar). Furthermore, there is no representation of the meaning of constituents, and, at least in these simple models, there are no features such as case, gender etc. So DOP primarily focuses on syntax, rather than semantics and morphology.

It is assumed that language users store all *fragments* of such representations, abstract as well as concrete, small as well as large, contiguous as well as discontiguous. The DOP-philosophy is:

(2.4) Since we do not know beforehand which [fragments] are important, we should not restrict them but take them all and let the statistics decide. (Bod, 2009, p. 755)

When producing new utterances, a *composition operation* combines stored fragments to form grammatical sentences. *Frequency* plays a crucial role here: fragments with a higher frequency are more 'imprinted' (more salient) and are thus more likely to be used in production.

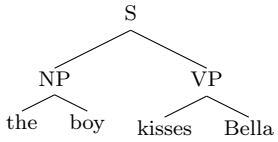
Formally, the fragments of language representation are *subtrees*, defined as follows:

Subtree a subtree of a tree T is a subgraph t of T such that

1. t consists of more than one node
2. t is connected
3. each non-frontier node in t has the same daughter nodes as the corresponding node in T

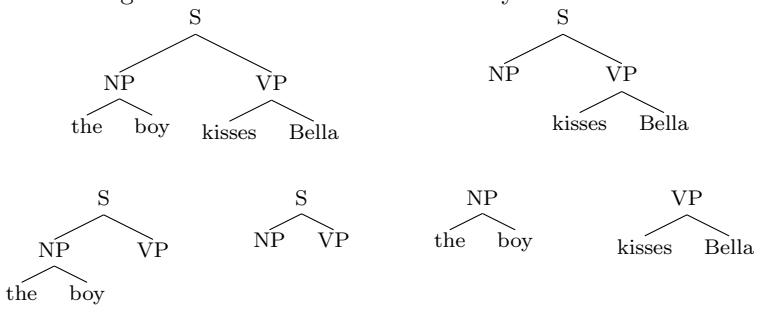
(Bod et al., 2003, p. 14)

For example, consider the following tree for the sentence 'the boy kisses Bella':



All subtrees for this tree (including the entire tree itself) are given in Figure 2.2.

Figure 2.2: All subtrees for 'the boy kisses Bella'



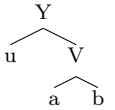
To generate new utterances, we define the following composition operation:

Composition the composition of tree t and tree u, written as $t \circ u$, is defined iff the label on the root node of u is identical to the label on the leftmost nonterminal leaf node of t. If $t \circ u$ is defined, it yields a copy of t in which a copy of u has been substituted on t's leftmost nonterminal node (Bod et al., 2003, p. 15)

For example, we can combine (2.5) and (2.6) to obtain (2.7).[2] However, (2.5) and (2.8) cannot be combined, because the label of (2.5)'s *leftmost* nonterminal (Y) is not equal to the label on the root node of (2.8) (Z).

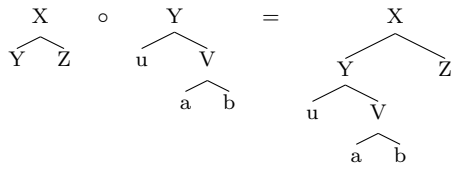
(2.5)

(2.6)



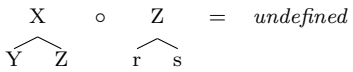
[2]Note that we indicate non-terminals with uppercase, and terminals with lowercase.

(2.7)



(2.8)

(2.9)



We stipulate that the composition operation is left-associative, so we can write $(t \circ u) \circ v$ as $t \circ u \circ v$. We define a *derivation* as a sequence of compositions.

The composition-operation allows us to generate trees for a given sentence; however, mostly there will be more than one possible tree. To disambiguate between different analyses, we calculate *probabilities* for trees —the tree with the highest probability will be considered the best analysis for the sentence.

First, we define the probability of subtrees, $P(t)$. Intuitively, we look at the relative probability of a subtree: the chance that we pick this subtree out of all subtrees we could pick (i.e. with the same root label). Let $|t|$ denote the frequency of subtree t, and $root(t)$ the label of the root node of subtree t. Then we define:

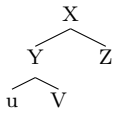
$$P(t) = \frac{|t|}{\sum_{t':root(t')=root(t)} |t'|}$$

Now, we can define the probability of a derivation, $P(t_1 \circ t_2 \circ \cdots \circ t_n)$. This is the joint probability of the subtrees t_1, ..., t_n. This simple model assumes that all subtrees are stochastically independent, so the joint probability is the product of the individual probabilities. Hence, we define:

$$P(t_1 \circ t_2 \circ \cdots \circ t_n) = \prod_{i=1}^{n} P(t_i)$$

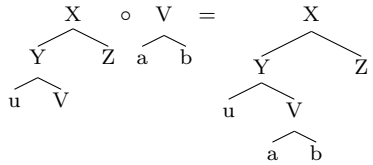
Finally, note that the same tree can be the result of several derivations. For example, (2.7) can be derived from (2.5) and (2.6), but also from (2.10) and (2.11).

(2.10)



(2.11)

(2.12)



The probability of a tree, $P(T)$, is then calculated by taking the sum of the probabilities of all its derivations (k is the number of derivations; n_j is the number of subtrees in derivation j; t_{ij} is the ith subtree of the jth derivation):

$$P(T) = \sum_{j=1}^{k} \prod_{i=1}^{n_j} P(t_{ij})$$

There are also versions of DOP where it is not the probability that disambiguates between different analyses, but the *length* of a derivation: the best tree is the tree that can be generated by the shortest derivation, viz. the derivation consisting of the least elements (cf. Bod (2000)). The idea behind this is that the shortest derivation will use larger fragments, i.e. larger syntactic contexts. Cognitively speaking, we can think of this as maximizing the structural analogy: for analyzing/producing new sentences, we try to maximize the similarity with previously encountered language experiences. Problematic, however, is that this shortest derivation may not be unique; in the case we resort to a frequency-based measure.

This frequency-based measure works as follows. We keep track of the frequencies of all subtrees. Then we can rank the subtrees according to their frequency —the subtree with the highest frequency will have rank 0, the second highest rank 1, etc. The idea is that the rank of a subtree indicates how many subtrees are preferable over that subtree. Of course, we will most likely also have ties in the ranking; for example, there will be many subtrees with frequency 1. In such a case, we will do the following. Suppose we have a subtree with frequency r, which has been assigned rank n, and that we now have s subtrees with frequency r', where r' is the largest number strictly less than r such that there are subtrees with frequency r'. Then we will assign all these subtrees rank $n + 1$. However, the next subtree(-group) in the ranking will be assigned rank $n + 1 + s$; after all, there are $n + 1 + s$ subtrees preferable over this subtree(-group). For example, suppose we have the frequencies in (2.13).

Then the ranking looks as in (2.14).

(2.13)

subtree	frequency
$s1$	3
$s2$	1
$s3$	1
$s4$	5
$s5$	3

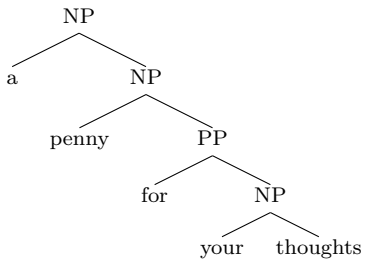
(2.14)

subtree	frequency	rank
$s4$	5	0
$s1$	3	1
$s5$	3	1
$s2$	1	3
$s3$	1	3

When we now have two equally short derivations, we can disambiguate between them by looking at the rank of their composing subtrees. The *ranking score* of a derivation is the sum of the rank of its subtrees. The derivation with the lowest ranking score is to be preferred. So a derivation consisting of $s1$ and $s2$ will have a ranking score of $1+3 = 4$; a derivation consisting of $s4$ and $s5$ will have a ranking score of $0 + 1 = 1$. Hence, the latter derivation will be preferred over the former. If multiple derivations have the same ranking score, we choose one randomly (although this will not often occur in real life).

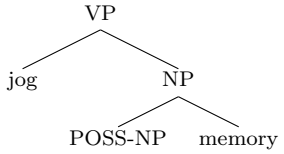
The simple DOP-model can be seen as a formal instantiation of the theoretical ideas in Usage-Based Grammar and Construction Grammar. In the first place, it is easy to see that subtrees formalize the notion of 'exemplars', in the sense that they are representations of 'constructions': expressions of different length and different levels of abstractness can be captured. For example, (2.15) can be considered a representation of construction (2.1), (2.16) of construction (2.2), and (2.17) of construction (2.3).[3]

(2.15)

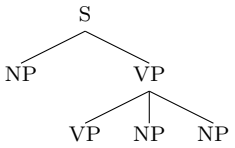


[3]Of course, I do not claim that these are the 'correct' trees, or that these are the 'correct' categories, but clearly DOP would propose some analyses along this line for these constructions.

(2.16)



(2.17)

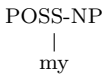


Note, however, that in Usage-Based Grammar exemplars are usually thought of as *complex* items, containing a range of information on a morphological, syntactic, semantic, ... level. In this simple DOP-model, the subtrees, which we consider to be exemplars, are clearly not complex items: they store syntactic information, and frequencies, but for example no semantic information. Later versions of DOP-models, such as LFG-DOP in Bod and Kaplan (2003), however, are capable of representing complex information.

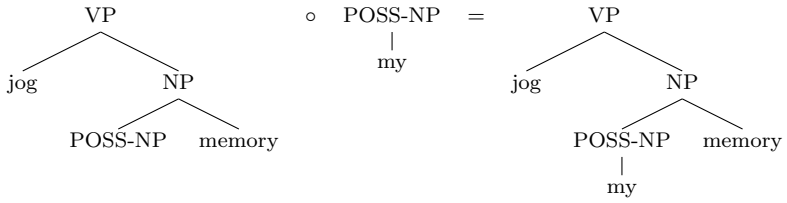
Second, the DOP-model is clearly usage-based. The grammar consists of the subtrees that are extracted from a corpus of language events. Hence, it emerges from experience. Since the grammar consists of subtrees, and subtrees are the representation of constructions, we can see that the DOP-model formalizes the conception of grammar as a 'construct-i-con', a repository of constructions.

Third, in Usage-Based Grammar and Construction Grammar new expressions are created by combining constructions; this is exactly what DOP's composition operation does. For example, the construction represented by the subtree in (2.16) can be combined with (2.18), as in (2.19).

(2.18)



(2.19)



Fourth, frequency plays an important role in DOP: to disambiguate between different analyses of a sentence, either probabilities or a ranking, which are both based on frequencies, are invoked.

Fifth, with respect to the distinction between the periphery and core of a language, DOP uses those mechanisms that are developed to account for the periphery (semi-idiosyncratic constructions are simply considered (fairly frequent) subtrees), to explain the core. This is exactly the methodology advised by Usage-Based Grammar and Construction Grammar.

Finally, DOP shares the view on language learning of Usage-Based Grammar and Construction Grammar. Also in the DOP-model there is no language-specific component hard-wired in the brain, but language is assumed to be learned via general cognitive capabilities, the most important of which is the ability to perform analogy: new structures are analyzed/produced by looking at similarities in previously encountered structures. Furthermore, also in the DOP-model the empirical input plays a key role: the grammar consists of subtrees which are extracted from concrete language experiences.

2.3 Unsupervised Data-Oriented Parsing

The version of DOP developed in the previous section is a theory of *supervised* parsing. This model parses (provides syntactic analyses of) sentences on the basis of a corpus with sentences that are already syntactically analyzed. Whereas this provides good results in a computational setting, especially when a large corpus with such annotations is at the researcher's disposal, a supervised parser is not an ideal model for language acquisition. After all, such a model of language also has to explain how children acquire the corpus of syntactically analyzed sentences in the first place. (But see Section 3.4 for further elaboration on this idea.)

For this reason, we turn towards *unsupervised* parsing. In unsupervised parsing, no preliminary information is assumed: the only input is a 'raw' corpus, containing sentences without any analysis at all.[4] It is safe to assume that also in language acquisition this is the only input the child receives. The task of an unsupervised parser is then to assign syntactic analyses to new sentences, on the basis of the raw corpus.

Following the general DOP-philosophy (cf. (2.4)), we think of unsupervised data-oriented parsing as follows:

> If a language learner does not know which phrase-structure tree should be assigned to a sentence, s/he initially allows for all possible trees and lets linguistic experience decide which is the "best" tree by maximizing structural analogy. (Bod, 2009, p. 760)

For an initial approximation, we do not consider *all* possible trees, but all *binary* possible trees. This is not a principled restriction, but it is merely made for computational efficiency. Furthermore, this is in line with the *Binary Branching*

[4] In most unsupervised parsers, however, it is assumed that the corpus has been preprocessed with a Part-of-Speech Tagger, which assigns a part-of-speech (Noun, Verb,...) to each word in the corpus. Also for the U-DOP parsers developed in this thesis, we will make this common assumption.

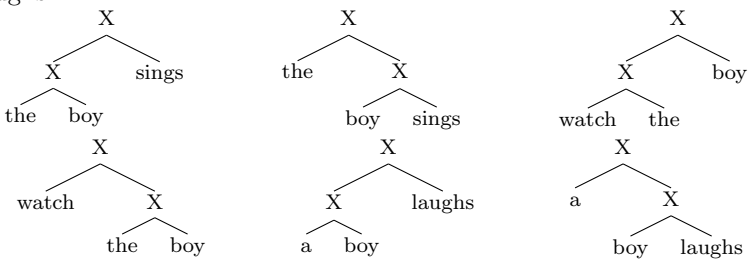
Hypothesis in Minimalist Grammar (cf. Subsection 4.1.3). In this initial version, we also do not assign category labels: every constituent is labeled 'X'. However, we will show how this restriction can be lifted in Section 3.3.

Intuitively, the parser works in three steps:

1. assign all trees to all sentences in the corpus
2. extract all subtrees from all trees
3. compute the best tree for a new sentence, on the basis of all subtrees

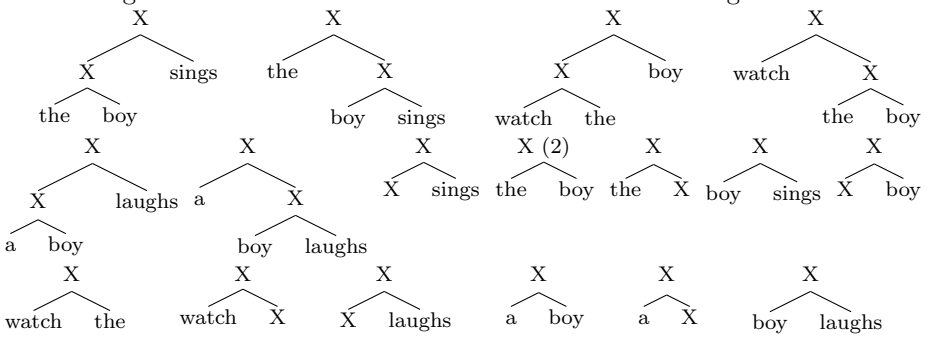
For example, suppose the corpus consists of the sentences 'the boy sings', 'watch the boy' and 'a boy laughs'. Then all possible (binary) trees are as in Figure 2.3. All subtrees that can be extracted from these trees are as in Figure 2.4. Note that the subtree '(X the boy)' occurs twice.

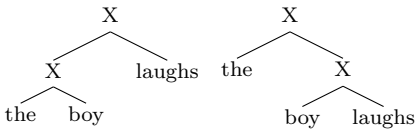
Figure 2.3: All trees for the sentences 'the boy sings', 'watch the boy' and 'a boy laughs'.



On the basis of these trees, we can now parse new sentences, in a similar way as in the DOP-models discussed in Section 2.2. For example, suppose we want to parse 'the boy laughs'. We see that there are two possible trees (the left tree is generally considered 'correct', the right tree 'wrong'):

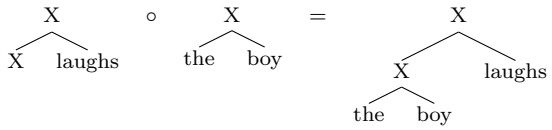
Figure 2.4: All subtrees extracted from the trees in Figure 2.3.



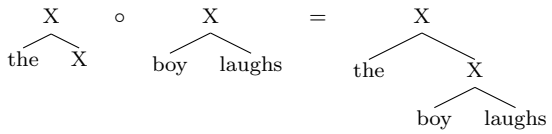


To disambiguate between the trees, we look at the measure of shortest derivation, as described in the previous section. Neither tree literally occurs in the corpus, so both trees have a shortest derivation with at least two elements. Indeed, we see that the shortest derivation is as in (2.20) (for the 'correct' tree) and (2.21) (for the 'wrong' tree):

(2.20)



(2.21)



We see that both derivations are equally short, and hence we must resort to the frequency-based measure. Since '(X the boy)' is the only subtree that occurs more than once, it gets rank 0; the other subtrees get rank 1. Hence, the rank for the first derivation is: 1 + 0 = 1; the rank for the second derivation is: 1 + 1 = 2. The first derivation has the lowest rank, and is hence preferable. So the parser correctly chooses the first tree as the syntactic analysis for the new sentence.

Cross-linguistically, the DOP-framework holds an entirely different view from generative grammar. In generative grammar, it is believed that all humans are endowed with innate parameters; differences across languages arise because the empirical input humans receive determines how the parameters should be set —and a parameter may be set differently in Japanese than in English. The DOP-framework, however, explains cross-linguistic differences in a different way. Since the grammar consists only of the fragments one encounters in actual language experiences, it is obvious that the grammar of a Japanese learner will be different from that of an English learner: the Japanese learner has encountered and hence stored different fragments, and will therefore produce different sentences.

Of course, a chicken-and-egg problem emerges here. What is left to explain, is how the variation in the input arises in the first place. However, this problem also arises for generative grammar: linguistic input is needed for the parameter setting, and the variation in this input is not fully explained. In the end, this

will need to be dealt with by theories of language evolution. See for example the work by Jelle Zuidema (e.g. Zuidema (2005)).

Another important difference with generative grammar lies in the class of sentences that is considered grammatical. In generative grammar, it is assumed that the grammar defines a well-circumscribed class of sentences that it can generate; these are grammatical, others are not. In the (U-)DOP-framework, however, in principle *all* sentences can be generated. This does not mean that all sentences are considered grammatical; rather, a different notion of grammaticality is used: *relative* rather than absolute grammaticality, i.e. some sentences are considered more grammatical than others. So a grammar produced in the (U-)DOP-framework does not define a class of grammatical sentences, but a distribution over sentences. This different conception of grammaticality leads to a certain amount of 'incommensurability': it is difficult to compare the two frameworks, since they differ in their concept of grammaticality. Therefore, a simplifying assumption has to be made to allow for a comparison: in the (U-)DOP framework, we only consider the sentences at the *top* of the distribution as grammatical. Then we can check whether generative grammar and (U-)DOP predict the same sentences to be grammatical.

Note that in principle language generation is the process from a semantic representation to a phonological representation, and language comprehension is the reverse. As mentioned earlier, the (U-)DOP models used in this thesis do not use semantics, and hence, are not rich enough to do the full task of language generation. However, it is possible to generate a *distribution* over sentences, and indicate which sentences are more likely to be generated than others.

Chapter 3

Implementations of U-DOP

In this chapter, our goal is twofold. In the first place, we want to build a concrete implementation of a U-DOP parser, following the theory outlined in the previous chapter. This concrete implementation will then be used in the next part to investigate whether the U-DOP framework can account for certain problems in language acquisition. In the second place, we want to improve existing parsing algorithms.

We will build four different parsing algorithms. The first algorithm is a fairly straightforward implementation of the basic ideas outlined in the previous chapter, although we will highlight some technical difficulties and their solutions. The second algorithm is an optimization of the first one, leading to a computationally more efficient parser, while accuracy results are not hurt too much. The third and fourth algorithm are real innovations which contribute new methods to the field of unsupervised parsing. The third algorithm is, to the best of our knowledge, the first parsing algorithm that merges the tasks of category labeling and parsing, hoping to improve both by the interaction. Finally, the fourth algorithm involves a new methodology which can simulate supervised parsing, while being in fact completely unsupervised, giving the best results on the ATIS-corpus reported so far. Moreover, this new methodology also has the fastest parsing times.

3.1 K-best shortest derivations with ranking as a second phase

In this section, we will develop our first parsing algorithm, a straightforward implementation of the shortest derivation – U-DOP theory outlined above. First, we will consider some technical problems (and their solutions) related to the training and parsing phase of the implementation. Next, we will discuss its evaluation.

3.1.1 The training phase

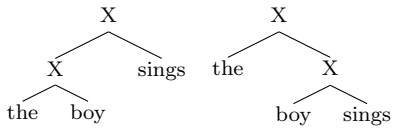
The first step in building a parser is to *train* on a *training corpus*. In U-DOP, this encompasses steps 1 and 2 of Section 2.3:

1. assign all trees to all sentences in the corpus
2. extract all subtrees from all trees

The task of the training phase is thus to build a *subtreebank*, containing all subtrees of all trees for all sentences in the training corpus; these subtrees can then subsequently be used in the parsing phase. Moreover, we must keep track of the frequencies of the subtrees, to build a *ranking* which can be used as a second phase to decide among the k-best derivations (cf. Section 2.2).

In practice, we do step 1 and 2 at the same time. We read the corpus sentence per sentence, and for each sentence we first compute all possible trees and then extract all subtrees from those possible trees, before moving on to the next sentence.

A first computational problem arises with the computation of all possible trees for a sentence: even though we make the simplification of working only with binary trees, the number of possible trees for a sentence still grows exponentially with the sentence length. This problem can be solved by storing all possible trees efficiently in a *chart*. For example, the two possible binary trees for the sentence 'the boy sings' are:



These can be stored efficiently in a chart as follows (the numbers indicate word indices: the element X in cell (i,j) represents a constituent/node in the tree spanning from word i to word j):

	0	1	2
0	the	X	X
1		boy	X
2			sings

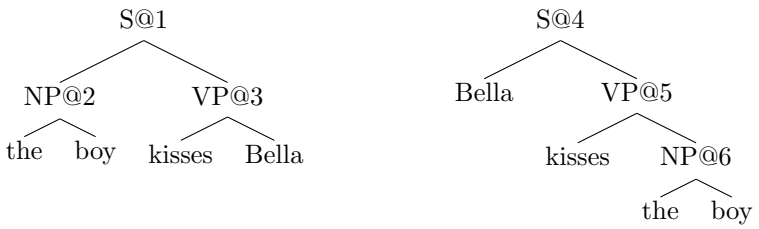
PCFG-reduction

The second computational problem is the number of subtrees that can be extracted from all these possible trees; this also grows exponentially with sentence length. The solution for this problem lies in the *Goodman PCFG-reduction* (Goodman, 2003). This PCFG-reduction was originally developed for supervised Data-Oriented Parsing, but it has been adapted to U-DOP in Bod (2007). In this thesis, we will develop a further reduction.

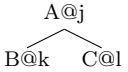
The original idea was that a derivation in DOP can be mimicked by a derivation with PCFG-rules. In the training phase, each node in each tree is assigned

a unique number, its address. Non-terminals with their address are called 'interior' nodes; non-terminals in their original form 'exterior' nodes. The idea is that interior nodes serve to "keep subtrees connected" after they are split into rules. For example, the trees for 'the boy kisses Bella' and 'Bella kisses the boy' then look as follows:[1]

(3.1)



Next, eight PCFG-rules are extracted for each node. Suppose we have a node such as

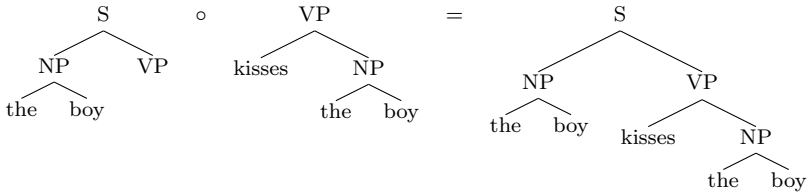


Then we generate the following rules (X_i is the nonterminal at address i):

$$\begin{array}{ll} A_j \rightarrow BC & A \rightarrow BC \\ A_j \rightarrow B_k C & A \rightarrow B_k C \\ A_j \rightarrow BC_l & A \rightarrow BC_l \\ A_j \rightarrow B_k C_l & A \rightarrow B_k C_l \end{array}$$

(We skip over the details of how to derive the probabilities for these rules; this is only necessary for probabilistic parsing, not for shortest derivation-parsing.) Now for each DOP-derivation, we can have a homomorphic PCFG derivation, which starts with an exterior non-terminal (without an address), then uses interior and exterior non-terminals for the intermediate steps, and ends with exterior (non-)terminals. For example, the tree for the sentence 'the boy kisses the boy' can be derived in the 'classic' DOP-way as in (3.2); the homomorphic PCFG derivation is given by applying the rules listed in (3.3) (note how the indices 'glue' together rules belonging to the same fragment in the derivation).

(3.2)

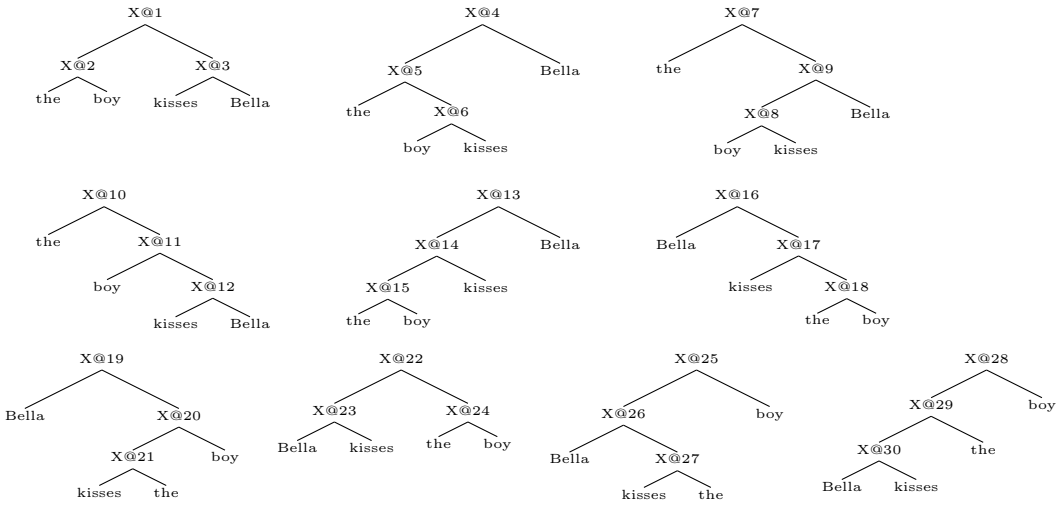


[1] Note that this is the method for supervised parsing; hence, there is a unique tree for each sentence, viz. the tree given in the training corpus.

(3.3)
1. S → NP@2 VP
2. NP@2 → the boy
3. VP → kisses NP@6
4. NP@6 → the boy

When we now move to unsupervised parsing, the naive idea would be that we assign unique addresses to each node in every tree for each sentence. However, this will quickly lead to an explosion of PCFG-rules. Therefore, Bod (2007) makes the simplifying assumption that we assign unique addresses to each node in the parse forest for each sentence. For example, the first approach would give rise to the trees and PCFG-rules in Figure 3.1 for the sentences 'the boy kisses Bella' and 'Bella kisses the boy' (88 rules). In contrast, the second approach leads to the parse forests and PCFG-rules in Figure 3.2 (70 rules). So for these two simple sentences we already see a large reduction in the amount of rules.

Figure 3.1: The trees and PCFG-rules for 'the boy kisses Bella' and 'Bella kisses the boy', according to the naive approach. Duplicate rules were deleted.



X@1 → X X	X → X X	X@16 → Bella X	X → Bella X
X@1 → X@2 X	X → X@2 X	X@16 → Bella X@17	X → Bella X@17
X@1 → X X@3	X → X X@3	X@17 → kisses X	X → kisses X
X@1 → X@2 X@3	X → X@2 X@3	X@17 → kisses X@18	X → kisses X@18
X@2 → the boy	X → the boy	X@18 → the boy	
X@3 → kisses Bella	X → kisses Bella	X@19 → Bella X	
X@4 → X Bella	X → X Bella	X@19 → Bella X@20	X → Bella X@20
X@4 → X@5 Bella	X → X@5 Bella	X@20 → X boy	X → X boy
X@5 → the X	X → the X	X@20 → X@21 boy	X → X@21 boy
X@5 → the X@6	X → the X@6	X@21 → kisses the	X → kisses the
X@6 → boy kisses	X → boy kisses	X@22 →X X	
X@7 → the X		X@22 → X@23 X	X →X@23 X
X@7 → the X@9	X → the X@9	X@22 → X X@24	X → X X@24
X@9 → X Bella		X@22 → X@23 X@24	X → X@23 X@24
X@9 → X@8 Bella	X → X@8 Bella	X@23 → Bella kisses	X → Bella kisses
X@8 → boy kisses		X@24 → the boy	
X@10 → the X		X@25 → X boy	
X@10 → the X@11	X → the X@11	X@25 → X@26 boy	X → X@26 boy
X@11 → boy X	X → boy X	X@26 → Bella X	
X@11 → boy X@12	X → boy X@12	X@26 → Bella X@27	X → Bella X@27
X@12 → kisses Bella		X@27 → kisses the	
X@13 → X Bella		X@28 → X boy	
X@13 → X@14 Bella	X → X@14 Bella	X@28 → X@29 boy	X → X@29 boy
X@14 → X kisses	X → X kisses	X@29 → X the	X → X the
X@14 → X@15 kisses	X → X@15 kisses	X@29 → X@30 the	X → X@30 the
X@15 → the boy		X@30 → Bella kisses	

In our implementation, we have even further reduced the amount of rules.[2,3] We follow Bod (2007) in assigning a unique address to each node in the parse forest for each sentence, but we assign the same address to nodes having the

[2]Thanks to Lorenz Demey for the inspiration for this idea.

[3]The approach adopted here is somewhat similar to the 'packed graph encoding' in Bansal and Klein (2010).

Figure 3.2: The parse forests and PCFG-rules for 'the boy kisses Bella' and 'Bella kisses the boy', according to the approach in Bod (2007). Duplicate rules were deleted.

	0	1	2	3
0	the	X@1	X@2	X@3
1		boy	X@4	X@5
2			kisses	X@6
3				Bella

	0	1	2	3
0	Bella	X@7	X@8	X@9
1		kisses	X@10	X@11
2			the	X@12
3				boy

X@1 → the boy	X→ the boy
X@4 → boy kisses	X→ boy kisses
X@6 → kisses Bella	X→ kisses Bella
X@2 → X kisses	X → X kisses
X@2 → X@1 kisses	X → X@1 kisses
X@2 → the X	X→ the X
X@2 → the X@4	X → the X@4
X@5 → boy X	X → boy X
X@5 → boy X@6	X → boy X@6
X@5 → X Bella	X → X Bella
X@5 → X@4 Bella	X → X@4 Bella
X@3 → X X	X → X X
X@3 → X@1 X	X → X@1 X
X@3 → X X@6	X → X X@6
X@3 → X@1 X@6	X → X@1 X@6
X@3 → X Bella	
X@3 → X@2 Bella	X → X@2 Bella
X@3 → the X	
X@3 → the X@5	X → the X@5

X@7 → Bella kisses	X → Bella kisses
X@10 → kisses the	X → kisses the
X@12 → the boy	
X@8 → X the	X → X the
X@8 → X@7 the	X → X@7 the
X@8 → Bella X	X → Bella X
X@8 → Bella X@10	X → Bella X@10
X@11 → kisses X	X → kisses X
X@11 → kisses X@12	X → kisses X@12
X@11 → X boy	X → X boy
X@11 → X@10 boy	X → X@10 boy
X@9 → X X	
X@9 → X@7 X	X → X@7 X
X@9 → X X@12	X → X X@12
X@9 → X@7 X@12	X → X@7 X@12
X@9 → X boy	
X@9 → X@8 boy	X → X@8 boy
X@9 → Bella X	
X@9 → Bella X@11	X → Bella X@11

same *yield*. For example, in the parse forests in Figure 3.2, nodes X@1 and X@12 have the same yield: they cover the same terminals, viz. 'the boy'. Therefore, we will assign them the same address, collapsing the PCFG-rules involving these nodes. More concretely, the parse forests and the PCFG-rules in our approach are in Figure 3.3. Of course, in this small example, the gain is rather small (we have 1 rule less than the approach of Bod (2007)). However, when considering a large corpus, where the same yields show up very often, the reduction gained can be quite significant: from 2,527,641 rules extracted from the child-directed speech of the Adam-part of the Childes corpus (Brown, 1973) to 914,744 rules. Finally, we note that a reduced PCFG not only improves the *memory* usage, but also the *speed* of the algorithm: fewer rules mean fewer possibilities to be considered.

The final thing we need to consider is the *weight* attributed to these rules. In a regular PCFG, rules are assigned probabilities. However, we will use this PCFG for parsing with the shortest derivation. Therefore, we will assign weight 1 to rules headed by an interior non-terminal, and weight 0.5 to rules headed

Figure 3.3: The parse forests and PCFG-rules for 'the boy kisses Bella' and 'Bella kisses the boy', according to the approach taken in this thesis. Duplicate rules were deleted.

	0	1	2	3
0	the	X@1	X@2	X@3
1		boy	X@4	X@5
2			kisses	X@6
3				Bella

	0	1	2	3
0	Bella	X@7	X@8	X@9
1		kisses	X@10	X@11
2			the	X@1
3				boy

X@1 → the boy	X→ the boy	X@7 → Bella kisses	X → Bella kisses
X@4 → boy kisses	X→ boy kisses	X@10 → kisses the	X → kisses the
X@6 → kisses Bella	X→ kisses Bella		
X@2 → X kisses	X → X kisses	X@8 → X the	X → X the
X@2 → X@1 kisses	X → X@1 kisses	X@8 → X@7 the	X → X@7 the
X@2 → the X	X→ the X	X@8 → Bella X	X → Bella X
X@2 → the X@4	X → the X@4	X@8 → Bella X@10	X → Bella X@10
X@5 → boy X	X → boy X	X@11 → kisses X	X → kisses X
X@5 → boy X@6	X → boy X@6	X@11 → kisses X@1	X → kisses X@1
X@5 → X Bella	X → X Bella	X@11 → X boy	X → X boy
X@5 → X@4 Bella	X → X@4 Bella	X@11 → X@10 boy	X → X@10 boy
X@3 → X X	X → X X	X@9 → X X	
X@3 → X@1 X	X → X@1 X	X@9 → X@7 X	X → X@7 X
X@3 → X X@6	X → X X@6	X@9 → X X@1	X → X X@1
X@3 → X@1 X@6	X → X@1 X@6	X@9 → X@7 X@1	X → X@7 X@1
X@3 → X Bella		X@9 → X boy	
X@3 → X@2 Bella	X → X@2 Bella	X@9 → X@8 boy	X → X@8 boy
X@3 → the X		X@9 → Bella X	
X@3 → the X@5	X → the X@5	X@9 → Bella X@11	X → Bella X@11

by an exterior non-terminal. Since we use weights instead of probabilities, we will henceforth speak of a CFG rather than a PCFG. The score of a derivation is obtained by multiplying the weights of the rules used (R_d is the set of rules used in derivation d):

$$\text{score}(d) = \prod_{r \in R_d} \text{weight}(r)$$

In this way, the derivation using these CFG-rules which maximizes the score will be the shortest derivation: using a rule with an interior non-terminal (i.e. building a subtree-element of the derivation) will not change the score, using a rule with an exterior non-terminal (i.e. starting a new element in the derivation) will lower the score. The fewer rules with exterior non-terminals are used (i.e. the fewer elements are introduced in the derivation), the better the score.

Ranking

The task of the training phase is to produce two objects which can be used in the parsing phase: a CFG and a ranking. The CFG will be used to form derivations, and the ranking as a second phase, to decide among the k-best derivations. The computational problems with the CFG are solved supra. A final problem is now the size of the ranking (see Section 2.2 for more information on how this ranking is built). This is a list with all subtrees extracted from the corpus, and their rank number, so this can become very large very fast. To reduce the size of this list, we are inspired by Zipf's law (cf. Manning and Schütze (1999, p. 23)): there are very few subtrees which occur frequently, and most subtrees occur only very rarely. More specifically, a large proportion of the subtrees on the list has frequency 1 (e.g. for the child-directed speech of the Adam corpus around 80%). Therefore, we replace this large amount of subtrees with just one line: the rank number of subtrees with frequency 1. When we now want to calculate the rank of a subtree in a derivation, we first check if it is present in the ranking. If it is, we use the frequency given by the ranking; if it is not, we assume that it has frequency 1, and assign it the rank specified in the additional line. Note that it is impossible that we encounter a subtree in a derivation that has in fact frequency 0 (i.e. it does not occur in the training corpus): after all, the subtrees in a derivation are built on the basis of the CFG-rules with unique nodes, extracted from the training corpus.

For example, the ranking in (2.14) will now look as follows:

(3.4)

subtree	frequency	rank
$s4$	5	0
$s1$	3	1
$s5$	3	1
subtrees with freq 1	1	3

Of course, in this small example, the gain is not so large: just one line. However, taking Zipf's law into account, the gain will be very large when the corpus is large, and there are many subtrees with frequency 1. For the child-directed speech from the Adam corpus, for example, the amount of subtrees in the ranking is reduced from 25,370,505 to 4,731,034.

3.1.2 The parsing phase

With the CFG and the ranking provided by the training phase, we can build a parser which provides syntactic analyses for new sentences. The parsing of a sentence occurs in two steps:

1. generate the k-best derivations
2. for each of the k best derivation, compute its rank, and take the derivation with the best (= lowest) rank

For the first step, we follow the third algorithm of Huang and Chiang (2005), the *lazy k-best algorithm*. In this algorithm, first a traditional Viterbi-chart

is created (cf. Manning and Schütze (1999, p. 396)), which numerates in an efficient way all possible subderivations. Next, the algorithm starts at the root node and recursively looks for the k-best derivations. A detailed explanation of the workings of this algorithm can be found in Brakel and Smets (2009).

When we have found the k-best derivations, we compute for each derivation its rank, by looking up the ranks of the composing subtrees, and taking the sum. Finally, the parser outputs the derivation with the lowest rank as its analysis for the sentence at hand.

3.1.3 Evaluation

For comparison with other approaches, we evaluate the parser on the Air Travel Information System (ATIS) corpus (Marcus et al., 1993). This small corpus contains around 500 utterances asking for travel information. Although most sentences are fairly small, it is still a relatively difficult corpus for parsing, because the utterances are spontaneous speech.

Following Brakel and Smets (2009), we split the ATIS corpus randomly ten times into 90% training corpus and 10% test corpus. In each iteration, we train the parser on the training corpus, and let the parser provide syntactic analyses for the test corpus (we set the parameter of the number of derivations to be considered (i.e. k) to 100, 10 and 1).[4] Next, we compared the analysis proposed by the parser with the gold standard: the manual annotation of the corpus.

From the gold standard, we ignored all productions that do not lead to a yield (following Brakel and Smets (2009)), and binarized the trees according to the procedure described in Goodman (1996). For all evaluation calculations, we used a self-written program that implements the evaluation measures used in Klein (2005).

Note that, following other research in this area, we only consider sentences of length 10 or less. Recall that we used part-of-speech tags rather than words as the raw corpus that serves as input. We look at the unlabeled precision, recall and F1-score (since all constituents simply have the label 'X', no labeled comparison with the gold standard is possible) and we take the average over the ten iterations. The scores are calculated as follows:

$$\text{Precision} \quad \frac{\text{number of correct constituents}}{\text{number of constituents proposed by the parser}}$$

$$\text{Recall} \quad \frac{\text{number of correct constituents}}{\text{number of constituents in the gold standard}}$$

$$\text{F1} \quad \frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}}$$

The results of the basic implementation developed in this section, together with a comparison with other approaches, can be found in Table 3.1.

To illustrate the benefits of taking all subtrees, of all depths rather than just depth 1 as in regular (P)CFG-parsing, we also evaluated the parser on the

[4] Obviously, for the setting $k = 1$, we do not rank the derivations in the second phase. If multiple derivations are equally short, we choose one randomly.

Table 3.1: Comparison of results. We compare with the systems EMILE and ABL from van Zaanen and Adriaans (2001), the algorithm described in Brakel and Smets (2009), the CDC-model proposed by Clark (2001) and the current state-of-the-art on the ATIS-corpus: the CCM-model developed in Klein (2005). We vary the parameter of the number of derivations to consider from 1 to 10 to 100. (Results better than the state-of-the-art are in bold.)

Previous approaches			
Model	R	P	F1
EMILE	16.8	51.6	25.4
B & S	29.0	29.0	29.0
ABL	35.6	43.6	39.2
CDC	34.6	53.4	42.0
CCM	47.6	55.4	51.2

Basic implementation				Version with short CFG			
Model	R	P	F1	Model	R	P	F1
basic100	36.4	36.7	36.6	basic100-short	28.5	28.7	28.6
basic10	**53.2**	**53.3**	**53.2**	basic10-short	36.3	36.7	36.5
basic1	**58.5**	**58.9**	**58.7**	**basic1-short**	**56.5**	**56.8**	**56.6**

ATIS corpus with another methodology (**short** in the table). From the CFG that is the result of the training phase, we delete all rules containing interior nodes (i.e. nodes with unique addresses, e.g. X@i). Since all interior nodes are deleted, every time a rule from this short CFG is used, a new element of the derivation is started. Hence, the derivation will only consist of elements which correspond to CFG-rules; for example, the element corresponding to the rule $X \rightarrow YZ$ is a subtree of depth 1:

(3.5)

As can be seen from Table 3.1, the score is worse with the **short**-CFG for each setting of the parameter (1, 10 and 100). Hence, we conclude that it is better to include subtrees of all depths. Therefore, parsing with all subtrees is superior over regular (P)CFG-parsing.

We see that the results improve the fewer derivations we consider. This indicates that the idea of using the shortest derivation is better than a frequency-based measure such as the ranking. However, it is remarkable that the basic1(-short) algorithm performs so well; after all, if there are multiple derivations that are equally short, this algorithm will pick a random one. Moreover, when used with the short-CFG, all derivations will be equally long (since only binary rules are used, all derivations will use the same number of rules), so the algorithm will always choose a random derivation, which makes it even more surprising that

it scores so well. I cannot say why this algorithm performs so well; in any case, it indicates that using maximal overlap/the shortest derivation as a criterion is better than using frequency/the ranking.

A comparison with earlier results shows that the basic implementation of shortest-derivation U-DOP outperforms the current state-of-the-art results on the ATIS corpus. Moreover, the CCM-model that produces this state-of-the-art was trained on the much larger Wall Street Journal corpus, whereas our models were only trained on 90% of the ATIS corpus. So we see that our models perform better even though they have access to less training data.

We would also like to evaluate the parser on the Wall Street Journal corpus, for better comparison with approaches such as Bod (2007). However, because this corpus is so large (around 7000 sentences), the CFG (and hence, the number of possibilities in a derivation) becomes so large that the parsing of new sentences takes too long to complete the evaluation.

3.2 Optimizing: looking just at the ranking

Since the previous algorithm is rather slow, we try to optimize it. The basic idea is that instead of using the ranking merely as a second phase *after* the best derivations have been calculated, we will use dynamic programming to calculate directly the derivation with the lowest rank.[5] The training phase is entirely similar to that described above. Below we describe the new parsing phase.

Intuitively, this algorithm stresses more the importance of *frequency* in language processing, rather than *maximal analogy*. Parsing with the shortest derivation will make use of large composing elements, which are often not so frequent. In contrast, this new algorithm will primarily use the most frequent building blocks, which are often not so large. Having these two parsers at our disposal, it will be interesting to see which disambiguation criterion will prove to be most succesful, in the first place at parsing sentences in general (w.r.t. F1-scores), but in the second place also at explaining the problems of language acquisition in Part II.

An example where this approach might perform better is when a sentence can be derived by either (i) one very rare tree (so the derivation consists of just one step, viz. this rare tree), or (ii) two very frequent subtrees. The first approach (of the previous section) will prefer the first derivation, whereas the second approach (of this section) will prefer the second: the rank of the rare tree will be very high, and the rank of the subtrees very low —so in sum, the rank of the second derivation will still be lower than that of the first, even though it consists of two composing elements (recall: the lower the rank, the better). However, in practice, it is difficult to find such a situation: most of the time the combination of the two frequent subtrees will *also* occur in the corpus; in that case, the shortest-derivation criterion will also choose the correct analysis,

[5] Thanks to Federico Sangati for the inspiration for this idea.

since it will use that combination of the two subtrees as the single element in the derivation.

So we expect the new algorithm to have a slightly worse F1-score than the previous algorithm. After all, it is possible that the derivation with the lowest rank is not included in the k-best shortest derivations. In this case, the previous algorithm will not output this 'wrong' tree, but this algorithm will, and hence, it will not output the 'correct' tree. Nevertheless, the main advantage of this new algorithm is that it is signifcantly faster than the previous one.

The optimized algorithm is summarized in Figure 3.4 (*numberOfWords* is the number of words in a sentence, e.g. for 'Edward kisses Bella' we have *numberOfWords*=3). First, we initialize the score of all terminals to 0. Then, we loop through the chart diagonal per diagonal, starting with the smallest constituents; e.g. constituents at diagonal 1 have a span of 2 (span 2 words), constituents at diagonal 2 have a span of 3, etc... Next, we look at each cell $(i,\ j)$ on the diagonal. Note that constituents in cell $(i,\ j)$ span from word i to word j. Then we consider all possible first-constituent – second-constituent combinations which lead to that span (e.g. first-constituent X with span $(i,\ k)$ and second-constituent Y with span $(k+1,\ j)$ where $i \leq k < j$). Now we look up in the CFG which nodes can be the head of those two constituents, i.e. all nodes X such that $X \rightarrow$ first-constituent second-constituent is a rule in the CFG. Note that we no longer use the weights attributed to the rules in the CFG; we merely use the CFG for listing the *possibilities*, no longer for choosing the (k-)best one(s).

Next, we look at each of those possible heads, and calculate the *score* for that head in that position. In the end, the top node with the lowest score will be the best derivation. The score of a head consists of three parts: the score of both its composing elements, and the *treeScore*. The purpose of the score is to keep track of the rank of the derivation so far (i.e. of the sum of the ranks of all subtrees used in the derivation so far). Initially, the score starts at 0. When the head is an interior non-terminal, we simply pass on the ranks of the composing elements, and define the *treeScore* to be 0, because the present element of the derivation is not yet finished. When an entire subtree of the derivation has been calculated (i.e. the head is an exterior non-terminal), the *treeScore* is the rank of that subtree. Through the dynamic programming this rank propagates up to the top node. Of course, we also store the *backtrace* for each element, which keeps track of the optimal path leading up to that element.

Table 3.2: Comparison of results, including the optimized algorithm. (Results better than the state-of-the-art are in bold.)

Model	R	P	F1
EMILE	51.6	16.8	25.4
B & S	29.0	29.0	29.0
ABL	43.6	35.6	39.2
CDC	53.4	34.6	42.0
optim-short	44.2	44.5	44.4
optim	48.3	49.0	48.7
CCM	55.4	47.6	51.2
basic1	**58.5**	**58.9**	**58.7**
basic1-short	**56.5**	**56.8**	**56.6**

Figure 3.4: The new algorithm, calculating the derivation with the lowest rank through dynamic programming.

for all elements in cells (i, i) (for $0 \leq i <$ *numberOfWords*) **do**
  score(*element*)=0
end for
for *diagonal* = 1 to *numberOfWords*-1 **do**
  for all cells (i, j) on *diagonal* **do**
    for all combinations (*firstConst*,*secondConst*) with span (i,j) **do**
      for all heads such that (*head* $\rightarrow$ *firstConst secondConst*) $\in$ CFG **do**
        newScore = score(*firstConst*) + score(*secondConst*) + *treeScore*
        if *newScore* < score(*head*) **then**
          score(*head*) = *newScore*
          backtrace(*head*) = (*firstConst*, *secondConst*)
        end if
      end for
    end for
  end for
end for

$$\text{treeScore} = \begin{cases} 0 & \text{if } \textit{head} \text{ is an interior node} \\ \text{ranking(subtree)} & \text{o/w} \end{cases}$$

For evaluation, we have performed the same procedure as in Section 3.1.3: 10 random train/test-splits of the ATIS corpus. The results, together with an overview of the results of other approaches, can be found in Table 3.2. Conform our expectations, this optimized algorithm performs worse than the basic imple-

Table 3.3: The time in seconds per sentence that each algorithm needs for training and parsing. All experiments for obtaining these results were conducted on an Acer Aspire M3710 desktop computer, with a Pentium Dual Core processor, 2.5GHz, 4GB RAM.

Model	Training	Parsing
basic1	0.14	26.44
basic10	0.14	46.94
basic100	0.14	97.17
optim	0.14	11.30

mentation, and even worse than the state-of-the-art. Nevertheless, it still has its fruitfulness due to its efficiency. The training time is of course the same as for the basic algorithm; parsing, however, takes much less time with the optimized algorithm, as can be seen from the data in Table 3.3.

For comparison, we also evaluated the version with the *short-CFG* methodology (cf. Section 3.1.3), to investigate whether it is also useful for this parser to include subtrees of all depths. Indeed, we find that the version with the short-CFG again performs a bit worse than the version which considers all subtrees.

3.3 Merging category labeling and parsing

In this section, we provide an extension to existing parsing methods. Up to now, unsupervised parsing has always been inherently *unlabeled* (see e.g. Bod (2007) and Klein (2005)). Additionally, some research has been done into *separate* labeling algorithms, which take the output of an unsupervised parser as input, and produce labels for the constituents that the unsupervised parser proposed (e.g. Reichart and Rappoport (2008) and Smets (2010)). The algorithm described here aims at doing the labeling and parsing simultaneously, in the hope that this will produce better results for both tasks.

The algorithm consists of 5 steps:

(1) First, we assign unique addresses to all nodes in the parse forest for each sentence, but the same address to nodes with the same yield (similar to our version of the Goodman (P)CFG-reduction in Section 3.1.1). So the nodes are X@1, X@2 ... Then we extract all subtrees from the parse forests of all sentences.

(2) In the second step, we apply techniques from Distributional Semantic Models (DSMs).[6] First, we build a co-occurrence matrix, with the nodes with their address (X@1, X@2 ...) as rows, and the extracted subtrees as columns. The value of cell (i, j) is then the number of times node i occurs in subtree

[6]For an introduction to DSMs, see Evert and Lenci (2009); for more details on the application of DSMs to syntactic category labeling, see Smets (2010, Section 4).

j. Next, we apply two dimensionality reductions. First, we set a limit to the subtrees that can occur as columns in the matrix: we only consider subtrees with a frequency greater than f (in the experiments, we set f to 100). Second, we apply Singular Value Decomposition (SVD), and reduce the matrix to s columns (in the experiments, we set s to 100). Finally, we perform k-means clustering on the reduced matrix, grouping the nodes into k groups (in the experiments, we set k to 2).[7]

(3) From the previous step, we know which group each node belongs to. In this step, we now replace each node X@i with Xℓ@i, where ℓ is the group node X@i belongs to according to the k-means analysis (so $1 \leq \ell \leq k$). So instead of nodes X@1, X@2, X@3 ... we now have nodes Xℓ@1, Xm@2, Xn@3 ... in the parse forests.

(4) Now we can proceed with training the parser as before (cf. Section 3.1.1). With respect to the CFG, the only difference is that instead of rules X@i $\rightarrow$ X@j X@s, we will now have rules Xk@i $\rightarrow$ Xℓ@j Xm@s. For the ranking, there is a greater difference. Rather than simply listing all subtrees in one ranking, we now have separate rankings for each syntactic category (i.e. a ranking for all subtrees headed by X1, another for all subtrees headed by X2, ..., another for all subtrees headed by Xk). Recall the intuitive idea behind the ranking: a subtree's rank reflects how many subtrees are preferred over it —and subtrees headed by a different syntactic category are incomparable.

(5) In the final step, we parse sentences in the same way as before (cf. Section 3.1.2). The resulting trees will now bear syntactic category labels X1, X2,...,Xk (with k the parameter for the k-means clustering). For a complete (labeled) comparison with a manually annotated gold standard, a final step is needed: a mapping between the proposed categories (X1, ..., Xk) and the categories in the gold standard (NP, VP, ...).

For example, suppose we have a training corpus consisting of the sentences 'the girl saw the boy', 'the boy saw the girl', 'the boy walked' and 'a girl walked'. The parse forests for the first step are:

	0	1	2	3	4
0	the	X@1	X@5	X@8	X@10
1		girl	X@2	X@6	X@9
2			saw	X@3	X@7
3				the	X@4
4					boy

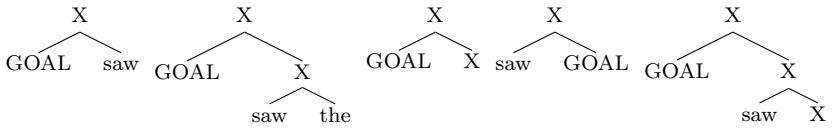
	0	1	2
0	the	X@4	X@19
1		boy	X@18
2			walked

[7] Note that k here signifies the number of labels/groups into which the data are clustered. This parameter must not be confused with the parameter used above the indicate the number of derivations to consider. (We refrained from using a different letter, to adhere to the standard terminology.)

	0	1	2	3	4
0	the	X@4	X@12	X@15	X@17
1		boy	X@11	X@13	X@16
2			saw	X@3	X@14
3				the	X@1
4					girl

	0	1	2
0	a	X@20	X@22
1		girl	X@21
2			walked

Now when we extract all subtrees and apply the techniques from Distributional Semantic Models, we will find that the constituents X@1 and X@4 are very similar: there are a lot of contexts in which both can occur, for example in the places marked with 'GOAL' in these subtrees:



Since these constituents have many contexts in common, the clustering algorithm will put them together in one category. So we can say that the algorithm has discovered the category 'Noun Phrase': constituents which can occur in the subject and the direct object position.

In the third step, we label each constituent in the parse forest with its cluster label, and the forests look as follows:

	0	1	2	3	4
0	the	X1@1	X2@5	X2@8	X2@10
1		girl	X2@2	X2@6	X2@9
2			saw	X2@3	X2@7
3				the	X1@4
4					boy

	0	1	2
0	the	X1@4	X2@19
1		boy	X2@18
2			walked

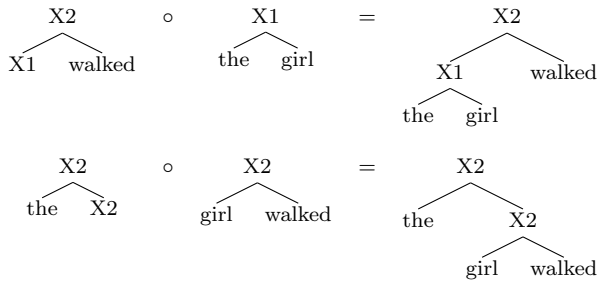
	0	1	2	3	4
0	the	X1@4	X2@12	X2@15	X2@17
1		boy	X2@11	X2@13	X2@16
2			saw	X2@3	X2@14
3				the	X1@1
4					girl

	0	1	2
0	a	X1@20	X2@22
1		girl	X2@21
2			walked

Now, suppose we want to find a derivation for 'the girl walked'. Since this sentence does not occur yet in the training corpus, the shortest derivation cannot consist of just one step: it minimally needs two composing elements. We find that the following two derivations are possible:

Table 3.4: The time in seconds per sentence that each algorithm needs for training and parsing. We report the time both for the algorithm with parsing only (cf. Table 3.3), and for the algorithm with labeling and parsing. All experiments for obtaining these results were conducted on an Acer Aspire M3710 desktop computer, with a Pentium Dual Core processor, 2.5GHz, 4GB RAM.

Previous algorithms			Labeling+Parsing		
Model	Training	Parsing	Model	Training	Parsing
basic1	0.14	26.44	basic1	7.34	25.72
basic10	0.14	46.94	basic10	7.34	46.35
basic100	0.14	97.17	basic100	7.34	132.46
optim	0.14	11.30	optim	7.34	10.30



Since both derivations are equally short, we need the ranking to decide which one is the best. The rank of the first derivation is $6 + 1 = 7$; the rank of the second derivation is $0 + 28 = 28$. Hence, we prefer the (correct) first derivation.

It is easy to see how the *parsing* phase of this parser will proceed a lot faster: fewer possibilities need to be considered. For example, if we have already established subderivations for X1 and X2, we only need to consider constituents that can be at the left-hand-side of the rule $\ldots \rightarrow$ X1 X2. If we did not have the labeling, we would have to consider all constituents which can be at the left-hand-side of the rule $\ldots \rightarrow$ X X, which would be a lot more. The *training* phase of this parser is still a bit inefficient: it takes a long time and a lot of memory (to build the co-occurrence matrix). This problem will definitely be taken up in further research. However, it is for an NLP-system still preferable to have a slow training phase (which in principle needs to be done only once), and a fast parsing phase (which will need to be done a lot), than vice versa. So in this respect this parser is definitely superior over the other two (in Sections 3.1 – 3.2).

In Table 3.4, we see how the training phase indeed takes much more time, when adding the labeling component, and that the parsing phase is slightly faster (except for the labeling+basic100-algorithm, where it takes longer than the version without labeling). Of course, on the small ATIS corpus the gains

Table 3.5: Results for experiments with the Labeling+Parsing-algorithm, compared with the results of earlier sections. We report the results both for the algorithm with parsing only (cf. Table 3.2), and for the algorithm with labeling and parsing. Reported are unlabeled recall, precision and F1-score. The parameters are: f = 100, s = 100, k = 2. (Results better than the state-of-the-art are in bold; cf. Table 3.1.)

Previous results of this thesis				10 rounds ATIS with Labeling+Parsing			
Model	R	P	F1	Model	R	P	F1
optim	48.3	49.0	48.7	optim	46.0	46.4	46.2
basic100	36.4	36.7	36.6	basic100	40.8	41.1	40.9
basic10	**53.2**	**53.3**	**53.2**	**basic10**	**52.9**	**53.9**	**53.4**
basic1	**58.5**	**58.9**	**58.7**	**basic1**	**56.0**	**56.3**	**56.2**

in parsing time are not so large —this difference will be more outspoken when considering larger corpora.

Also the possibility to compare the labeled bracketing of the parser with the gold standard bracketing is a large contribution: other parsing algorithms can only compare unlabeled bracketing; other labeling algorithms can compare labels, but for the bracketing they need to rely on other parsers.

The final question is whether this parser is also superior with respect to F1-scores: our claim is that the incorporation of a labeling algorithm not only allows for labeled bracketing at all, but also improves the quality of the unlabeled bracketing, i.e. the bracketing benefits from the labeling. Previous unsupervised algorithms could not check this, because all existing labeling algorithms rely on an unsupervised parser in the first place.

Problematic for this algorithm is that it needs quite a large training corpus to get the data for the second step: we need much co-occurrence information in order to get the clustering right. For this reason, separate labeling algorithms such as Smets (2010) were evaluated on the basis of the Wall Street Journal corpus —and on this large corpus they got quite good results. However, evaluation on such a large corpus is not yet feasible with the algorithm described here: even though we have dimensionality reduction techniques to reduce the size of the co-occurrence matrix, we still need to extract all subtrees of all possible trees and keep track of their frequency, which leads to memory issues when using a large corpus. Therefore, we were only able to evaluate the algorithm on the basis of the small ATIS corpus; moreover, it only made sense to cluster into two categories: for more categories, we would need more data. The results can be found in Table 3.5. We only look at unlabeled bracketing results, so we only look whether this algorithm gets better *parsing* results; to see whether it also gets better *labeling* results, we would have to cluster into 26 categories, so that

we can compare with the gold standard —and we would need much more data to be able to cluster into 26 categories.

We see that the results of the Labeling+Parsing algorithm are no improvement of the results of the parsing algorithm on its own; in fact, in some cases the results even deteriorate. We believe this is due to the fact that there are not enough data available to do the labeling right —and with a flawed labeling, also the bracketing results will be worse. However, we still think that in principle the addition of the labeling component will also improve the (unlabeled) bracketing results —only, we would need to be able to evaluate on a larger corpus to test this. Therefore, further research should certainly concentrate on optimizing the algorithm so that this hypothesis can be further investigated.

3.4 A new approach to unsupervised parsing

In this section, we will present another innovation to the field of unsupervised parsing. The innovation lies not so much in the parsing techniques themselves, but rather in the methodology: we will simulate supervised parsing, while still assuming no richer input than an unannotated corpus of text.[8]

Current U-DOP parsers have one flaw: they take *all* subtrees into account, so also the 'flawed' ones. For example, every time the parser encounters in its training phase a sentence containing the sequence 'noun coordinating-conjunction' (e.g. in the sentence 'Bella loves wolves and vampires'), the subtree (X noun coordinating-conjunction) is extracted, even though this subtree does not occur in the 'correct' parse of the tree. Of course, U-DOP's main argument is that the frequency will rule out 'flawed' subtrees. However, this argument does not apply to this case: the flawed subtree (X noun coordinating-conjunction) will be fairly frequent, because it is extracted every time a conjunction is encountered.

This inclusion of 'flawed' subtrees is countered effectively by a *supervised* DOP-parser, since such a parser only extracts subtrees from the 'correct' trees. With such a parser, subtrees like (X noun coordinating-conjunction) will not be extracted, or only with a very low frequency. However, using a supervised parser would not say much about language acquisition. The question remains: how do children acquire the training corpus of (correctly) parsed trees in the first place? Since we want to model (certain aspects of) language acquisition in this thesis, we cannot make use of a supervised parser.

Therefore, we still start from a corpus that could be the input for any unsupervised parser, i.e. without syntactic annotations. It is safe to assume that this is the input children receive. Hence, a parser assuming only this input is adequate for modeling language acquisition (in contrast with a supervised parser). Next, we *simulate* supervised learning on the basis of this corpus in the following way.[9]

[8] Although we do require part-of-speech tags, like most current unsupervised parsers.

[9] This approach is in many aspects similar to the one in Bod (2007). However, whereas we use *all* correct subtrees, Bod uses only those subtrees used in the derivations. Though probably more fruitful in the long run, the latter approach requires much larger training

The algorithm consists of five steps:

(1) First, we divide the training corpus into two parts: an Extraction Corpus (EC) and a Held-out Corpus (HC). For now, we follow Bod (2007) in splitting the training corpus in half: 50% EC and 50% HC. However, it is possible that other divisions might be more fruitful.

(2) In the second step, we train an unsupervised parser on the EC. This can be any of the unsupervised parsers discussed above, or even other unsupervised parsers such as those of Bod (2007) or Klein (2005).

(3) Third, we parse the HC on the basis of this parser trained on the EC. Again, parsing can proceed as with any of the parsers discussed above, or another unsupervised parser. The output of this step is a syntactically annotated corpus that can serve as input for a supervised parser.

(4) In the fourth step, we use the parsed HC as the annotated training corpus for a supervised parser. Note that again, in principle all supervised parsers can be used in this step. Our proposal is a simple adaptation of the unsupervised parsers developed in Sections 3.1 – 3.2: rather than taking all subtrees from all possible trees, we now only extract those subtrees of the 'correct' trees, i.e. of the trees the unsupervised parser in step 3 proposed. Parsing then proceeds in the same way as above.

(5) Finally, we can use the supervised parser trained in step 4 to parse new test sentences. In our proposal, this parsing algorithm will be identical to that of the parsers in Sections 3.1 – 3.2.

The problem mentioned in the beginning of this section will be solved with this methodology: the expectation is that subtrees like (X noun coordinating-conjunction) won't occur very often in the parsed HC, and hence have a very high rank. Therefore, the (supervised) parser trained on the parsed HC will prefer other subtrees (with a better ranking score) over this one.[10]

With respect to efficiency, we see the same scheme as with the labeling parser from Section 3.3: the training phase takes quite long, but the parsing

corpora, and hence will not be further explored in this thesis.

[10]Of course, it is possible to debate what exactly the 'correct' tree for a coordination structure is. However, speaking from a (U-)DOP point-of-view, we believe a strong argument can be made for the analysis proposed here, viz. (X noun (X coordinating-conjunction noun)) rather than (X (X noun coordinating-conjunction) noun). After all, a language user will never encounter sentences ending with a sequence 'noun coordinating-conjunction', so the subtree (X noun coordinating-conjunction) cannot occur at the end of a sentence. In contrast, she *will* encounter sentences ending with the sequence 'coordinating-conjunction noun', and hence the subtree (X coordinating-conjunction noun) can occur at the end of a sentence. So based on the EC (X noun coordinating-conjunction) and (X coordinating-conjunction noun) are equally likely to occur; hence, a parser trained on the EC alone would be able to give a pretty good score to sentences like 'what do you love chicken and' (it could use the frequent subtree (X noun coordinating-conjunction). However, in the HC, sentences such as 'what do you love chicken and' will not occur, whereas sentences like 'you love chicken and what' will occur. Therefore, in the second phase (after training on the parsed HC), the subtree (X coordinating-conjunction noun) will be more frequent than the subtree (X noun coordinating conjunction). So the crucial idea is that, contrary to normal unsupervised parsing, we also assume that the HC, the corpus to be parsed, contains 'grammatical' sentences, and that these will improve the parsing results. Since this thesis is not about providing *correct* analyses of structures, but rather about modeling *relative* grammaticality, we won't go into more detail here.

Table 3.6: The time in seconds per sentence that each algorithm needs for training and parsing. We report the time both for the algorithm without the new methodology (cf. Table 3.3), and with the new methodology. All experiments for obtaining these results were conducted on an Acer Aspire M3710 desktop computer, with a Pentium Dual Core processor, 2.5GHz, 4GB RAM.

Previous algorithms			New methodology		
Model	Training	Parsing	Model	Training	Parsing
basic1	0.14	26.44	basic1	4.24	0.02
basic10	0.14	46.94	basic10	7.76	0.04
basic100	0.14	97.17	basic100	28.38	36.13
optim	0.14	11.30	optim	1.96	0.04

of new sentences goes very fast. For NLP applications, this is what we want, since training only needs to be done once, and parsing all the time. The training phase comprises steps 1 – 4; the parsing phase step 5. The bottleneck in training lies in Step 3: the parsing of the HC. This still takes long, because it is based on the quite large CFG extracted from the EC. In contrast, the parsing of new sentences in step 5 is based on the CFG extracted from the 'correct' trees produced in step 4, which is much smaller.

In Table 3.6, we see the training and parsing times for the basic and optimized algorithms, first without the new methodology, and then with the new methodology. We see that the training takes a bit longer, but that the gain in parsing time is huge.[11] Contrary to the Labeling+Parsing-algorithm, we see already a significant gain on the small ATIS corpus.

Another disadvantage of this approach might be that one would expect it to need a large training corpus. After all, since the parses for the sentences in the HC form the basis for the final parser, we want these parses to be as good as possible —hence, the training corpus on which their parses are based (the EC) should be as large as possible. However, the EC is only 50% of the original training corpus. A solution to this problem is the *leave-one-out-methodology*:[12] instead of dividing the training corpus into two separate halves, the EC and the HC, we let the EC be the whole corpus, and the HC one sentence that is removed from the EC —this procedure is repeated for each sentence. So, for example, if the training corpus has 500 sentences, we parse the first sentence (= HC) on the basis of sentences 2 – 500 (= EC); then we parse the second sentence (= HC) on the basis of sentences 1, 3 – 500 (= EC) and so on until we parse the last sentence

[11] Only the basic100-algorithm still takes surprisingly long to parse. The reason for this is that, after the listing of the possibilities, this algorithm still has to compute the 100-best derivations —and this still takes some time. So we still see a great increase in time efficiency, but the advantage is less outspoken than with the other algorithms, because it is overshadowed by the second step in parsing, i.e. the extraction of the 100-best derivations.

[12] Thanks to Federico Sangati for this suggestion.

(= HC) on the basis of sentences 1 – 499. The training phase of this procedure can be done efficiently by first training on all sentences (all 500 in the example), and then removing at each iteration the rules for the sentence in the HC (for example, all rules extracted from sentence 1 in the first iteration). Although this procedure deals with the sparse data problem (the training corpus used as the basis for the parses now only contains one sentence less than the original training corpus), it suffers greatly in speed: because all sentences need to be parsed, rather than just half of the sentences in the corpus, the time needed for step 3 is greatly increased —and step 3 was already the slowest step in the algorithm. Considering the results on the ATIS corpus, however, we needn't worry about data sparsity issues: even on this small corpus we obtain results that are better than the state-of-the-art.

For evaluation, we first applied the same methodology as in the previous sections: 10 rounds of train/test-splits on the ATIS corpus;[13] the results can be found in Table 3.7. However, because of the slowness of Step 3 of the algorithm, it was impossible to complete the evaluation for the algorithm using the (slower) basic implementation; using the (faster) optimized implementation (optim), we do obtain results. We see that this algorithm achieves better results with the new methodology: we gain 5% in F1-score. This is in accordance with our general claim that this new methodology achieves better results with any unsupervised parser.

To further investigate the possibilities of this new methodology, we also evaluated it with the inferior version of the basic implementation, i.e. with the short-CFG (containing only subtrees of depth 1). Also in this case, we see that the version with the new methodology systematically performs better than the old version, for each setting of the parameter (1, 10 and 100). Since we already know that the version with the short-CFG always performs worse than the 'full' version (cf. the results in Table 3.1), this gives a promising outlook on the results with the new methodology and the full CFG – these will most likely be even better. This hope is further nourished by the difference between the results of the new methodology with the optimized algorithm and the full CFG, and the new methodology with the optimized algorithm and the short-CFG: also in this case, the version with the full CFG performs vastly better.

We also evaluated the methodology with the leave-one-out solution (cf. Table 3.8). However, it was so slow that we could only obtain results for the new methodology with the optimized algorithm. Somewhat surprisingly, we see that the results are slightly worse with this solution.

Finally, it is also possible practically to apply this methodology to parse the ATIS corpus on the basis of training on the Wall Street Journal corpus, but only when using a short-CFG in Steps 2 and 3 (cf. Table 3.9). Note that these results are directly comparable with those of the state-of-the-art (Klein, 2005): also Klein trained on the WSJ and tested on ATIS. We proceeded as follows. **(1)** We divided the WSJ into 50% EC and 50% HC (using the leave-one-out

[13] So for every experiment round, we take 45% of the whole corpus for the EC, 45% for the HC and 10% as a test corpus.

Table 3.7: Results for experiments with the new methodology, compared with the results of earlier sections. (Results better than the state-of-the-art are in bold; cf. Table 3.1.)

Previous results of this thesis				10 rounds ATIS with new methodology			
Model	R	P	F1	Model	R	P	F1
optim	48.3	49.0	48.7	**optim**	**51.7**	**56.2**	**53.9**
optim-short	44.2	44.5	44.4	optim-short	40.36	45.53	42.71
basic100-short	28.5	28.7	28.6	basic100-short	46.4	66.1	49.8
basic10-short	36.3	36.7	36.5	**basic10-short**	**48.1**	**65.1**	**55.3**
basic1-short	**56.5**	**56.8**	**56.6**	**basic1-short**	**52.0**	**65.6**	**58.0**

methodology was not feasible, because this would involve parsing the entire WSJ, which could not be completed in time). **(2)** We extract all subtrees from the EC using the methodology described in Subsection 3.1.1, which is the training phase of both the basic and the optimized algorithm. **(3)** We parse HC using the short version of a parser. **(4)** We extract all subtrees from the parsed HC. **(5)** We parse the ATIS corpus using the full version of the parser used in step 3; so for the line 'optim' in Table 3.9, we used optim-short in step 3 and (full) optim in step 5; for the line 'basic1', we used basic1-short in step 3 and (full) basic1 in step 5, etc.

We see that the results are better when we train on this larger corpus. Again, this provides a promising perspective: the results with the full CFG will most likely be even better. Interestingly, the differences between the optimized and the basic implementation with various parameters when training on such a large corpus disappear. More specifically, it are the *trees* that the algorithms produce that are identical, not the *derivations*. We think that this is due to the fact that the short version of the parsers is used in step 3, and the fact that we train on already parsed trees (= the output of step 4). Both facts greatly reduce the amount of rules and hence the amount of possibilities: therefore, the ambiguity as to which is the correct tree is greatly reduced. In any case, the fact that the optimized algorithm performs as good as the basic implementation provides a promising perspective, since the former is much more efficient than the latter.

3.5 Summary

In this chapter, we have built four implementations of a U-DOP inspired unsupervised parser. Already the first, straightforward implementation of the basic U-DOP ideas performed better than the current state-of-the-art on the ATIS corpus. However, this implementation was very slow and inefficient, so we developed an optimization of it. The scores of this optimization were slightly worse,

Table 3.8: Results for experiments with the new methodology, using the leave-one-out methodology. (Results better than the state-of-the-art are in bold; cf. Table 3.1.)

10 rounds ATIS with leave-one-out			
Model	R	P	F1
optim	**50.0**	**55.0**	**52.4**

Table 3.9: Results for experiments with the new methodology, using the WSJ as training corpus. (Results better than the state-of-the-art are in bold; cf. Table 3.1.)

Parse ATIS based on WSJ with new methodology			
Model	R	P	F1
optim	**58.6**	**64.1**	**61.2**
basic100	**58.6**	**64.1**	**61.2**
basic10	**58.6**	**64.1**	**61.2**
basic1	**58.6**	**64.1**	**61.2**

but the gain in space and time efficiency was signficant.

Next, we developed two innovations to the field of unsupervised parsing. First, we built an algorithm that is the first one to do the tasks of labeling and parsing simultaneously. The hypothesis is that both tasks will benefit from this interaction. However, we were unable to test this hypothesis up to now, because the training corpus needs to be quite large for the labeling component, which brings along some memory issues. Second, we developed a new methodology for unsupervised parsing, which can in principle be used by any unsupervised parser. In this chapter, we have applied it succesfully to the parsers developed in Sections 3.1 and 3.2. Because of computational issues, we could only conduct experiments with the weaker versions of the parsers, i.e. with a short-CFG (not considering all subtrees). However, already these weaker versions showed impressive results, which provide a promising outlook for use with the 'full' version of the parsers. Although its training phase is a bit slower than that of the other implementations, its parsing phase is by far the fastest. This is exactly what we want from an NLP-application: training (which needs only be done once) may be a bit slow, as long as the parsing (which needs to be done often) goes fast. Interestingly, this new methodology both gives the best F1-scores, and the best parsing times: it is both the fastest *and* the best performing algorithm proposed in this thesis.

The results we have obtained so far are thus already better than the current

state-of-the-art, and we see a great potential for even better results. However, to obtain these better results, we need to be able to evaluate on larger corpora, and with the full versions of the parsers. Therefore, the first thing further research should focus on is to optimize the existing parsers, so that space and time issues become less pressing. Second, the delicate interplay between maximal analogy (shortest derivation) and frequency (ranking) as disambiguation criteria needs to be further investigated. It is clear that both criteria should play a role, but how much influence each criterion has needs to be further worked out. Looking at the F1-scores, we would predict that maximal analogy should play a bigger role than frequency: the implementations based on the shortest derivation achieve higher scores than the implementations based on the ranking.

The primary goal of this thesis, however, is to explain problems of language acquisition without resorting to nativist hypotheses, postulating innate constraints and principles. Now that we have working implementations of the U-DOP framework, we can use these in the next part to model how children can acquire certain constructions in an empirical way.

Part II

Typical problems in language acquisition

After having introduced our U-DOP theory and implementation of language acquisition, we will now look at various problems of language acquisition. These are constructions of which nativist linguists typically maintain that it is impossible that children could acquire them on the basis of evidence only (e.g. Crain and Thornton (2006)); they claim that certain principles have to be innate in order to account for the fact that children can learn the (un)grammaticality of these constructions.

First, we will give an introduction to the current state-of-the-art in the nativist linguistic tradition: Minimalist Grammar. We will introduce all principles and machinery necessary to understand a minimalist account of the constructions under investigation.

Next, we will look at certain constructions in more detail. In the first place, we will look at all details of wh-questions. According to nativists, wh-questions can be accounted for by a coherent set of constraints, viz. the island constraints. We will try to account for this construction in an empirical way in the simple framework of DOP. Note that we do not claim that these constraints are wrong, or that they cannot be innate; rather, we refute the nativist argument that the constraints *have* to be innate, by showing that a purely empiricist explanation based on general structure-induction techniques is possible.

In the second place, we look at related phenomena, that can be explained by the same constraints, and show that the U-DOP approach is at least as general as the nativist approach in that it can also account for these phenomena. Moreover, we also discuss phenomena that are not explainable by the constraints, but that our U-DOP approach can explain, thus showing that the U-DOP approach is more general.

Chapter 4

The Basics of Minimalist Grammar

In this chapter we will give a short introduction to the theoretical framework of Minimalist Grammar (MG), mainly based on Adger (2003). We will introduce all aspects of the framework necessary for developing a minimalist account of the phenomena discussed in the further chapters. The reader who is acquainted with MG can skip this chapter. In the next chapters we will then compare the minimalist account of the constructions with the explanation emerging from the U-DOP theory developed in Part I.

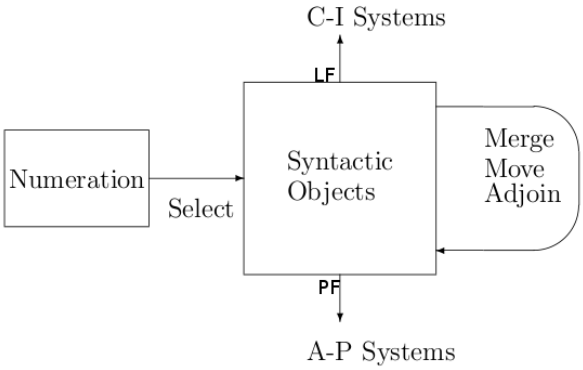
4.1 System architecture and basic elements

In this section, we will describe the general system architecture of the Minimalist Grammar framework, and explain the basic elements on which the system builds. In the next section, we will use these elements to build the main structure of sentences.

4.1.1 System architecture

The main components of the system's architecture are represented in Figure 4.1. The principal component is the *syntactic component*: it contains syntactic objects on which syntactic operations such as Merge, Move and Adjoin can be performed (cf. Subsections 4.1.3 and 4.1.5). The input for the syntactic component is the *numeration*, or the lexical component: a collection of lexical items. The interface between this component and the syntactic component is grounded in the 'Select' relation: a syntactic derivation starts off by selecting lexical items from the numeration. The syntactic component also interfaces with the semantic component: the *Conceptual-Intentional system* (C-I system). The syntactic component outputs a *Logical Form* (LF), on which interface rules can apply to link the syntax with the semantics. For example, to account for

Figure 4.1: the basic components of the minimalist system



the fact that morphological inflections influence meaning, there is an interface rule such as (4.1) which applies to the LF that the syntactic component outputs and feeds into the C-I system.

(4.1) Interpret a noun specified with [number:plural] as referring to a group of entities.

Finally, the syntactic component also interfaces with the *Articulatory-Perceptual system* (A-P system). To interface with this component, the syntactic component outputs a *Phonetic Form* (PF), which is the result of *Spellout*. Spellout is a set of rules, such as (4.2), to transform a syntactic object into a representation that can interface with the AP-system.

(4.2) Pronounce v[~~uInfl~~:past] as *ed*

4.1.2 Features

The smallest unit in the syntactic component is the lexical item, which is selected from the numeration. Lexical items are specified with *features*:

Feature a morphosyntactic feature is a property of words that the syntax is sensitive to and which may determine the particular shape a word has (Adger, 2003, p. 24)

Features can be either *privative* or *valued*. Valued features can have different values; for example, the feature 'number' can have the value 'singular' or 'plural'. This is represented as in (4.3). Privative features have no value; they are either present or they are not; for example, the feature 'noun' is present on nouns, but not on other lexical items. This is represented as in (4.4).

(4.3) (a) [number:singular]

(b) [number:plural]

(4.4) [N]

A further distinction can be made between *interpretable* and *uninterpretable* features. Interpretable features have an effect on the semantic representation. For example, the 'number' feature indicates whether we are talking about one object, or more than one. Uninterpretable features, on the other hand, do not affect the semantic representation; these are purely syntactic in nature. For example, the 'case' feature only influences the syntax. We will indicate that a feature is uninterpretable by putting a u in front of it, e.g. [$u$case:] is the uninterpretable feature 'case'. Since interpretable features influence the semantic representation, they have to be accessible at the interface between the syntactic and the semantic component (the LF); rule (4.1), for example, uses the interpretable 'number'-feature. Uninterpretable features, however, must be eliminated before the semantic rules apply. This principle is called 'Full Interpretation':

Full Interpretation the structure to which the semantic interface rules apply contains no uninterpretable features (Adger, 2003, p. 85)

This elimination of uninterpretable features is caused by the syntactic operations.

4.1.3 Trees and Merge

In MG, syntactic objects always take on the form of a tree. Out of two syntactic objects (incl. lexical items), a new tree can be constructed via the operation *Merge*. First, note that each syntactic object has a *label*; usually, this label is formed out of the features of the object.[1] Now suppose that we have two syntactic objects with labels 'X' and 'Y'. Then Merge can create a new syntactic object with label 'Z' by combining 'X' and 'Y':

We say that Z *immediately contains* X and Y. The usual terminology of *nodes*, *mothers*, *daughters*, *root* and *terminal nodes* applies. Like we said, Merge can construct a tree out of *two* syntactic objects; in MG, Merge is not allowed to create unary or n-ary branching trees (for $n > 2$). This is called the *Binary Branching Hypothesis*.

A more detailed definition of Merge will be discussed in the next subsection, after we have introduced some other necessary concepts.

So far, the Merge operation seems very similar to the operations in U-DOP (cf. Section 2.2). However, there are some differences. First, Merge does not distinguish between (4.5) and (4.6): it does not specify the linear order of the

[1] E.g. an object with feature [N] will have as label 'N'.

elements it merges, whereas in (U-)DOP (4.5) and (4.6) are two distinct subtrees. Of course, natural language does have a fixed word order (at least in English) —this is taken care of in MG by linearizing the structures.[2]

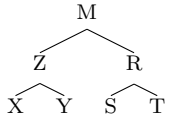
(4.5)

(4.6)

Second, Merge only combines objects at their root nodes. So we can Merge (4.5) and (4.7) to obtain (4.8), but we cannot insert one constituent inside another: we cannot Merge (4.5) and (4.9) to obtain (4.10). Note, however, that we would be able to combine (4.5) and (4.9) in the U-DOP formalism.

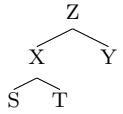
(4.7)

(4.8)



(4.9)

(4.10)



4.1.4 Theta-roles and checking

MG distinguishes between *arguments* and *adjuncts*. Arguments are constituents that are obligatory for the grammaticality of a sentence (e.g. (4.11) is grammatical, but (4.12) is not, because the argument 'Edward' is missing). Adjuncts are optional: they can be left out without influencing the grammaticality of the sentence (e.g. both (4.11) and (4.13) are grammatical, even though the adjunct 'every day' is left out in (4.11)).

[2]How structures are linearized is explained in the next subsection.

(4.11) Bella loves Edward.

(4.12) * Bella loves.

(4.13) Bella loves Edward every day.

We say that *predicates subcategorize* for certain kinds of expressions. These expressions can be of different semantic types. For example, in (4.14) the predicate ('give') subcategorizes for an Agent (something that initiates the action, in casu 'Bella'), a Theme (something that undergoes the action, in casu 'the book') and a Goal (something towards which the action is oriented, in casu 'Edward'). Agent, Theme and Goal (among others) are called *thematic roles* or θ-roles.

(4.14) Bella gives the book to Edward

So a predicate assigns θ-roles to constituents in a sentence. More specifically, a predicate has to assign each θ-role it has to a constituent in a sentence, otherwise the sentence is ungrammatical (e.g. (4.12) is ungrammatical, because the predicate's Theme-role is not assigned). Moreover, each constituent in the sentence can only be assigned one θ-role (e.g. we cannot say that in (4.12) 'Bella' is both the Agent and the Theme, meaning that she loves herself). These two observations taken together constitute the Unique θ-Generalization:

Unique θ-Generalization each θ-role must be assigned but a constituent cannot be assigned more than one θ-role (Adger, 2003, p. 81)

The apparatus to enforce this generalization consists of *categorial selectional features* (c-selectional features).

C-selectional features a c-selectional feature is a categorial feature on a lexical item, which does not determine the distribution of the lexical item itself; rather it determines the category of the elements which will be able to Merge with that lexical item

For example, the predicate 'love' carries two c-selectional features, [uN,uN],[3] indicating that it can Merge with two items which have themselves the categorial feature [N] (indicating that they are nouns). Recall the distinction made above between interpretable and uninterpretable features. Clearly, these c-selectional features have a purely syntactic function and are thus uninterpretable. Therefore, we mark them with a u, to distinguish them from the interpretable categorial features. Above we mentioned the principle of Full Interpretation: uninterpretable features must be deleted before LF. Now we have all the machinery in place to establish this.

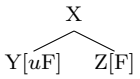
We have the following two principles:

[3]Recall that u in front of a feature indicates that it is uninterpretable. So the 'N'-feature on 'love' is uninterpretable, because it indicates which category of elements will be able to Merge with 'love', but the 'N'-feature on 'Edward' is interpretable, because it indicates the category of 'Edward' itself.

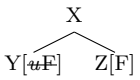
Checking Requirement uninterpretable (c-selectional) features must be checked, and once checked, they can delete

Checking under Sisterhood an uninterpretable c-selectional feature F on a syntactic object Y is checked when Y is sister to another syntactic object Z which bears a matching feature F

Suppose we have the following tree:

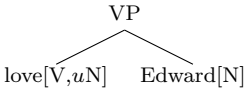


Y has an uninterpretable feature F, which needs to be checked according to the Checking Requirement. Z bears a matching feature F, and is in a sisterhood relation with Y, so by Checking under Sisterhood Y's [uF] can be checked:

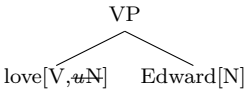


At the point where LF will be produced, all checked features will delete, and Full Interpretation will be satisfied.

A more concrete example is the following. Suppose we have the lexical items *love[V,uN]* and *Edward[N]*. We can Merge these two items and get



Now we can check the uninterpretable uN-feature and get



Note that we need a constituent with a matching feature for all c-selectional features on a predicate. The link with the Unique θ-Generalization is now fairly straightforward: each θ-role is associated with a c-selectional feature and hence, each θ-role will be associated with a constituent in the sentence. A remaining problem is how to determine which constituent gets which θ-role; this is called the *Linking Problem* and will be dealt with in Subsection 4.2.1.

Now we see that Merge is triggered by feature checking: we need to Merge with other constituents to be able to check uninterpretable features. Then we have the following definition of Merge:

Merge

1. Merge applies to two syntactic objects to form a new syntactic object
2. the new syntactic object is said to contain the original syntactic objects, which are sisters but which are not linearized
3. Merge only applies to the root nodes of syntactic objects

4. Merge allows the checking of an uninterpretable c-selectional feature on a head, since it creates a sisterhood syntactic relation

(Adger, 2003, p. 90–91)

This is the most specific definition given in Adger (2003). When we look at Chomsky (1995), the definition proposed there is not much clearer:

> Applied to two objects α and β, Merge forms the new object K, eliminating α and β [...] K must be constituted somehow from the items α and β [...] The simplest object constructed from α and β is the set $\{\alpha, \beta\}$, so we take K to involve at least this set, where α and β are the *constituents* of K [...] K must therefore at least (and we assume at most) be of the form $\{\gamma, \{\alpha, \beta\}\}$, where γ identifies the type to which K belongs, indicating its relevant properties. Call γ the *label* of K. (Chomsky, 1995, p. 243)

Although both definitions are not mathematically precise, we can gather what is meant. Consider trees as graphs, i.e. tuples (V, E) such that $E \subseteq V \times V$, with E being the set of edges, and V the set of vertices. Now suppose we have two graphs/trees A and B, with $\alpha \in V(A)$ the root node of A, and $\beta \in V(B)$ the root node of B. Furthermore, let H be a vertex such that $H \notin V(A) \cup V(B)$. Then we define the Merge of A and B as follows:

$$\mathbf{Merge}_H(A, B) =: M \text{ such that}$$

$$\begin{cases} V(M) = \{H\} \cup V(A) \cup V(B) \\ E(M) = \{(H, \alpha), (H, \beta)\} \cup E(A) \cup E(B) \end{cases}$$

So if A and B are trees with heads α and β, then $\mathbf{Merge}_H(A, B)$ is a tree with head H, which contains A and B (and has α and β as daughters).

Note that this definition keeps intact the property of Merge to not specify a linear order, i.e. to not distinguish between (4.5) and (4.6).

Now we can define *heads* in an independent way:

Head the head is the syntactic object which selects in any Merge operation (Adger, 2003, p. 91)

An important property of heads is that they *project*: when a head is merged with another constituent, its features are projected to the new syntactic object. We distinguish between *minimal*, *intermediate* and *maximal* projections. The minimal projection is the lexical item itself (designated e.g. 'N'). When all c-selectional features of a head are satisfied, we call it a maximal projection (designated e.g. 'NP'). Note that only maximal projections can be merged with other heads (otherwise the unchecked c-selectional features would remain unchecked and the derivation would eventually crash because the Checking Requirement would not be satisfied). With an intermediate or bar-level projection, there are still some c-selectional features to check (designated e.g. '$\overline{\text{N}}$'). The sister of a bar-level projection is called a *specifier*.

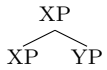
Now we can say how structures are linearized (recall that Merge doesn't specify a linear order). In English, heads come to the left of the complements they select; specifiers, however, occur to the left of their head. Adjuncts, finally, can appear either to the left or the right of the phrase they adjoin to.

4.1.5 Other syntactic operations

Recall the difference between arguments and adjuncts made above. In the previous section we have shown how Merge is triggered by the need for c-selectional feature checking. So arguments are incorporated in a sentence via the Merge-operation. Adjuncts, however, are entirely optional —so there are no c-selectional features triggering their Merge into the sentence. Therefore, a new syntactic operation *Adjoin* is defined, which is not triggered, but can be applied to every maximal projection.

Adjoin inserts a phrasal object into another phrasal object at its outermost level. It does not create a new object, it expands one of the old ones by stretching its outermost layer into two parts and inserting the adjoined object between them (Adger, 2003, p. 112)

For example, in the following tree YP is adjoined to XP:



So the resulting object still has all the features of the phrase to which the adjunct is adjoined.

The final operation that can be performed on syntactic objects is *Move*.

Move takes a structure formed by applications of Merge, and then moves one of the elements of that structure into another position in the tree (Adger, 2003, p. 132)

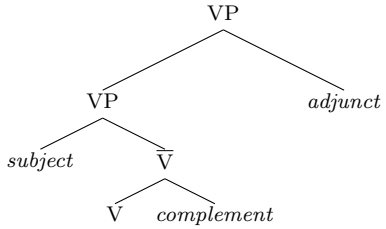
We will see examples of the workings of Move in the next section.

4.2 Basic structure of sentences

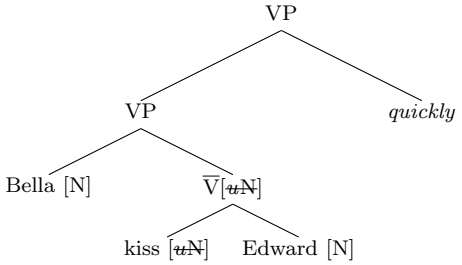
In this section, we will gradually build the basic structure of sentences. We start with the structure of the VP, gradually enlarging our scope to the TP and finally to the CP.

4.2.1 Structure of the VP

Intuitively, we would build the VP as follows:

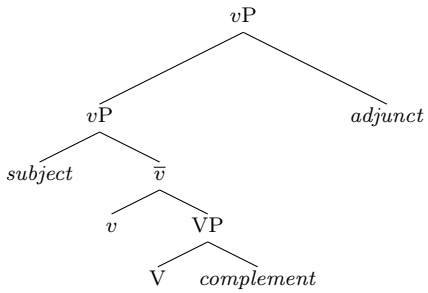


For example, the structure of 'Bella kisses Edward quickly' would be:



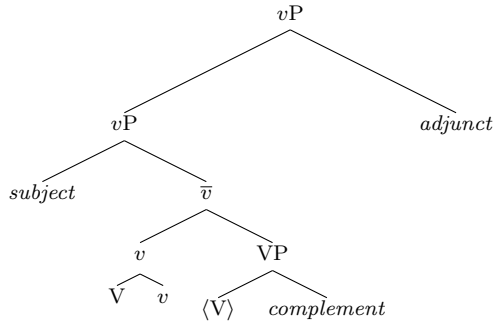
We see that Full Interpretation is satisfied. The predicate 'kiss' has two c-selectional features. The first one is checked by 'Edward'; the second one is projected to the bar-level projection, where it is checked by 'Bella'.

However, in the minimalist Framework this simple structure is rejected, building on an argumentation making use of ditransitives (for more details, see (Adger, 2003, Section 4.4)). An additional layer called v ('little v') is introduced. The basic structure of the verb phrase is thus as follows:[4]

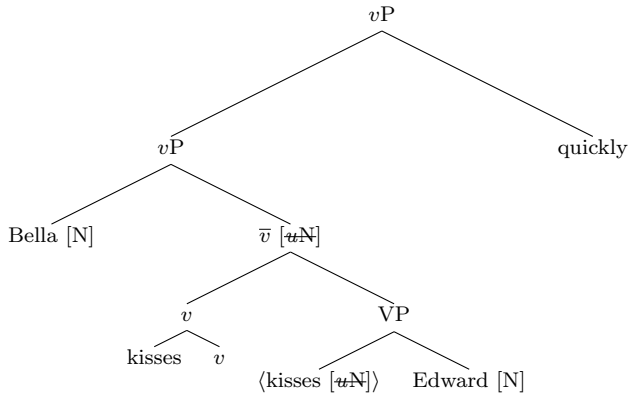


Again, based on an argumentation making use of ditransitives, the operation Move will apply to the verb V, and Move it to v. The resulting structure is (note that V has adjoined with v, and the angled brackets indicate the trace of V):

[4]The c-selectional feature that selects the subject is now no longer on V, but on v.



So the structure of 'Bella kisses Edward quickly' is:



We see that the verb 'kiss' only has one c-selectional feature, which is checked by 'Edward'. The new constituent v bears an additional c-selectional feature, which is projected up to the bar-level projection, where it is satisfied by 'Bella'. All uninterpretable features are checked, and Full Interpretation is satisfied.

The question remains why the VP merges with v. Since v doesn't assign a θ-role to the VP, we cannot argue that Merge is triggered by c-selectional features in this case. Rather, a new mechanism is introduced: the *Hierarchy of Projections*. For now, the Hierarchy of Projections takes the form of (4.15); we will amend it in later sections.

(4.15) $v > \mathrm{V}$

This hierarchy states that what is left of the >-sign always has to merge with what is right of the >-sign. So here we stipulate that v will always merge with V.

Now we are also able to solve the linking problem: the machinery already ensures that each θ-role is associated with a unique constituent, but how do we know with which constituent a θ-role is linked? The answer lies in the *Uniformity of θ-Assignment Hypothesis* (UTAH):

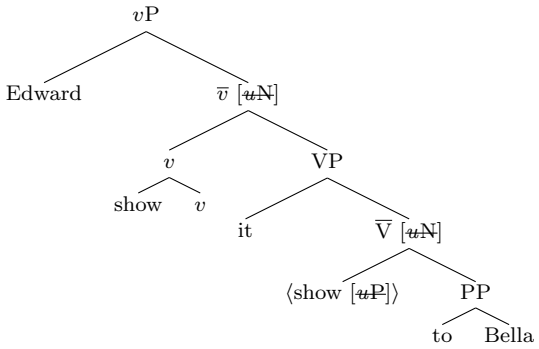
UTAH identical thematic relationships between predicates and their arguments are represented syntactically by identical structural relationships

when items are Merged (Adger, 2003, p. 138)

So by the structural position of a constituent, we know with which θ-role it can be linked. Following the trees given above, it is claimed that

- an NP that is the daughter of a vP is interpreted as an Agent
- an NP that is the daughter of a VP is interpreted as a Theme
- a PP that is the daughter of $\overline{\text{V}}$ is interpreted as a Goal

For example, the sentence 'Edward shows it to Bella' has the following structure:



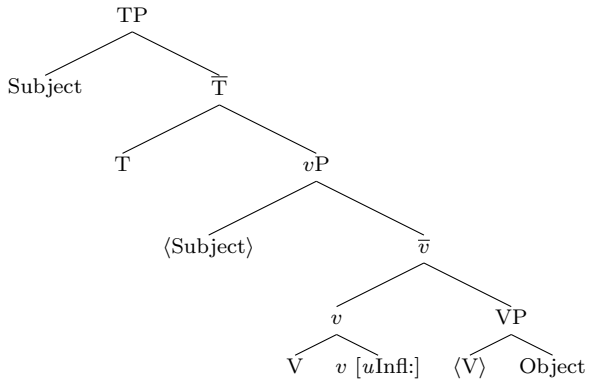
Now we see that 'Edward' is the daughter of a vP and hence interpreted as the Agent; 'it' is the daughter of a VP and interpreted as the Theme; and finally, 'to Bella' is the daughter of $\overline{\text{V}}$, and interpeted as the Goal.

4.2.2 Structure of the TP

In Minimalist Grammar, it is argued that there is an additional layer on top of the vP: the TP. The minimal projection of the TP, the category T, hosts the *tense* features for the whole sentence. The argumentation for introducing this extra layer is based on modals and VP-ellipsis. Again, we need the Hierarchy of Projections to explain why Merge between T and vP takes place; the adapted (but not yet final) version is in (4.16):

(4.16) T > v > V

The general structure of the TP is the following:



We see that the verb has moved to v, as before, but now also the Subject has moved, to the specifier of TP. This is to account for the fact that subjects appear to the left of modals. We also see another new technicality: the feature [uInfl:]. Recall that we made a distinction between privative and valued features. Up to now, we have only encountered privative features (the c-selectional features). [uInfl:] is a valued feature; it doesn't have a value yet, but it will be assigned a value simultaneously with its checking. For features of this kind, we don't apply the Checking under Sisterhood-principle (as we did for c-selectional features), but we define a new kind of relation, *Agree*:

Agree in a configuration X[F:val] $\cdots$ Y[uF:] where $\cdots$ represents c-command, then F checks and values uF, resulting in X[F:val] $\cdots$ Y[~~uF~~:val] (Adger, 2003, p. 169)

So according to the Agree relation, an uninterpretable feature gets a value and is checked at the same time, and it doesn't have to be in a sisterhood relation with a matching feature —c-command suffices.[5] This relation holds between T and v to ensure that the tense features that T hosts will be pronounced on the verb. Furthermore, there is an additional condition on the Agree-relation, which will become important later on:

Locality of Matching Agree holds between a feature F on X and a matching feature F on Y if and only if there is no intervening Z[F] (Adger, 2003, p. 238)

With intervention defined as follows:

Intervention in a structure [X $\cdots$ Z $\cdots$ Y], Z intervenes between X and Y iff X c-commands Z and Z c-commands Y (Adger, 2003, p. 238)

Auxiliaries are said to have a category of their own (Perf, Prog), which can optionally be merged with a vP. When we also consider negation (Neg) and the passive (Pass), the Hierarchy of Projections looks as follows (the brackets indicate that the Merge is optional):

[5] C-command is defined as follows: "a node A c-commands a node B if, and only if A's sister either: (a) is B, or (b) contains B" (Adger, 2003, p. 117).

(4.17) T > (Neg) > (Perf) > (Prog) > (Pass) > v > V

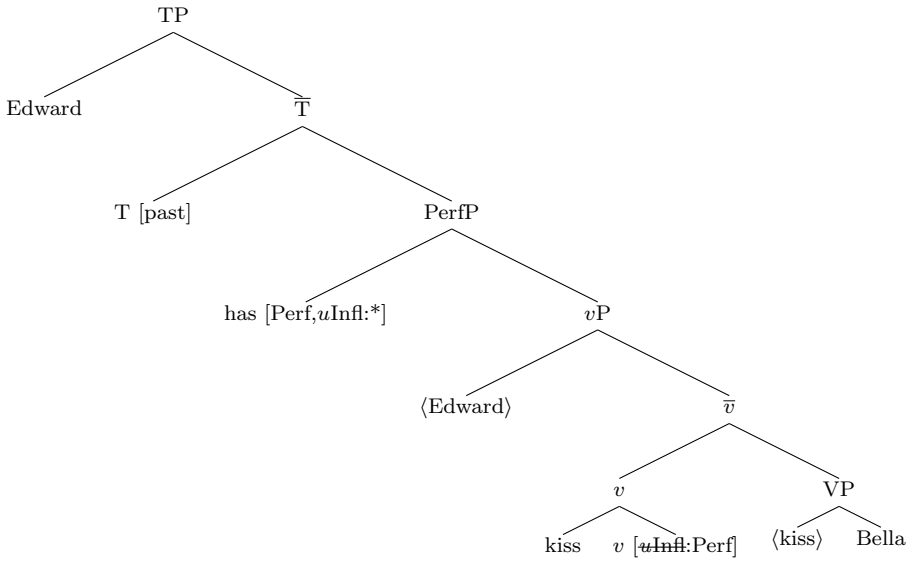
Based on the position of negation (after the auxiliary), it is claimed that the auxiliary must move to T, to end up higher than the negation. To explain what triggers this movement, a new technicality is introduced: feature strength.

Feature strength a strong feature must be local [i.e. in a sisterhood relation] to the feature it checks/is checked by (Adger, 2003, p. 179)

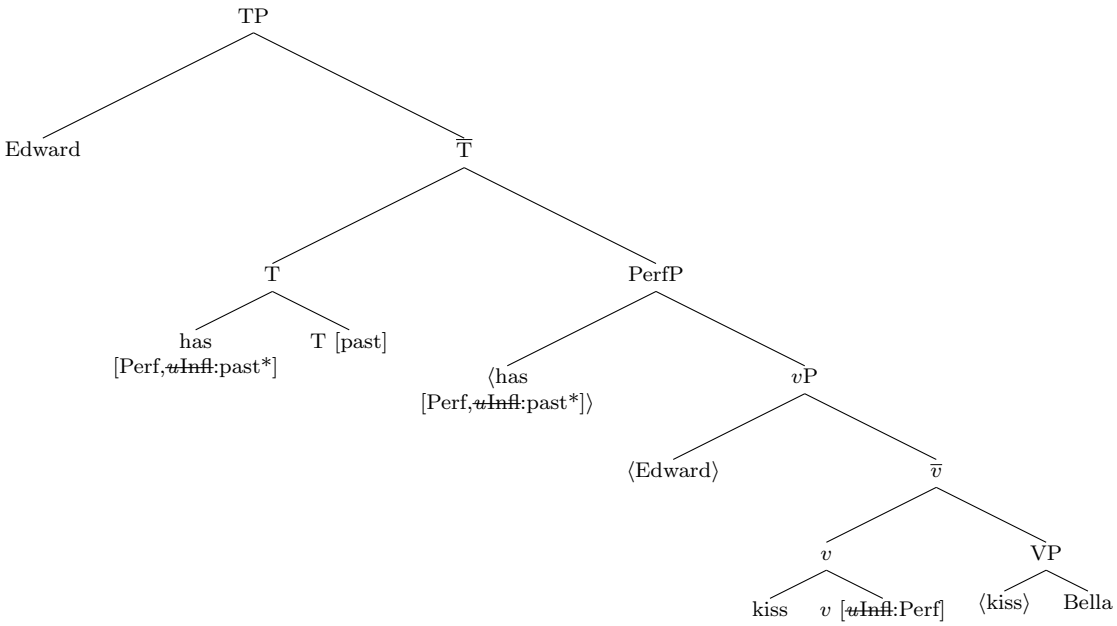
We can represent this schematically as in (4.18). X bears an uninterpretable strong feature F, which needs to be checked. Since X is in a c-command relationship with Y, Agree can take place between X and Y. However, since F is a strong feature, Y has to be in a local relationship; hence, Y is Moved.

(4.18) X[uF*] $\cdots$ Y[F] $\longrightarrow$ X[~~uF*~~]Y[F] $\cdots$ $\langle$Y[F]$\rangle$

Now we can account for the movement of an auxiliary to T. We say that the [uInfl:] feature on the auxiliary is strong. T checks and values this feature, and since it is strong, it has to move to ensure that it is in a sisterhood relation with T. For example, the structure of the sentence 'Edward has kissed Bella' before movement is as follows (the Perf-feature on the auxiliary checks and values the (weak) uInfl-feature on v):



Now, in order to check the strong uInfl-feature on the auxiliary, it has to Agree with T, and because the feature is strong, the auxiliary must move to a position where it is a sister of T.



4.2.3 Structure of the CP

The final layer we add to the structure of a sentence is the CP. The difference between the TP and the CP is that the former deals with tense, aspect and other properties of the event picked out by the verb, whereas the latter indicates how the hearer should think of the proposition expressed by the clause. For example, a complementizer such as 'that' indicates that the proposition is considered a fact, whereas a complementizer such as 'whether' indicates that the proposition is a question about facts.

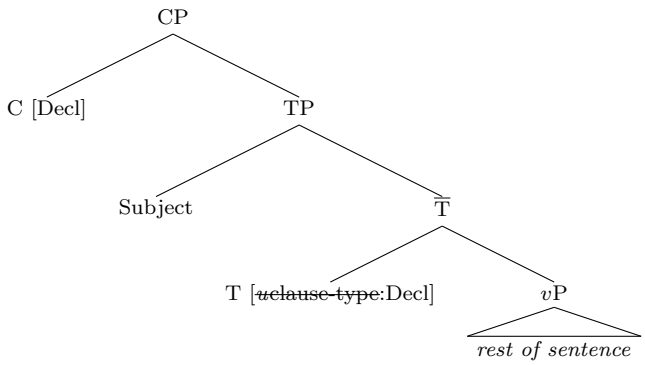
To enforce Merge between the CP and the TP, we adapt the Hierarchy of Projections to its final version:

(4.19) C > T > (Neg) > (Perf) > (Prog) > (Pass) > v > V

Like T hosts the tense-features, C hosts a clause-type-feature, with values [Q] (question) and [Decl] (declarative). This feature is interpretable on C, and hence does not need to be checked. However, this feature can check and value

an uninterpretable clause-type-feature on T. So we have as structure:

(4.20)

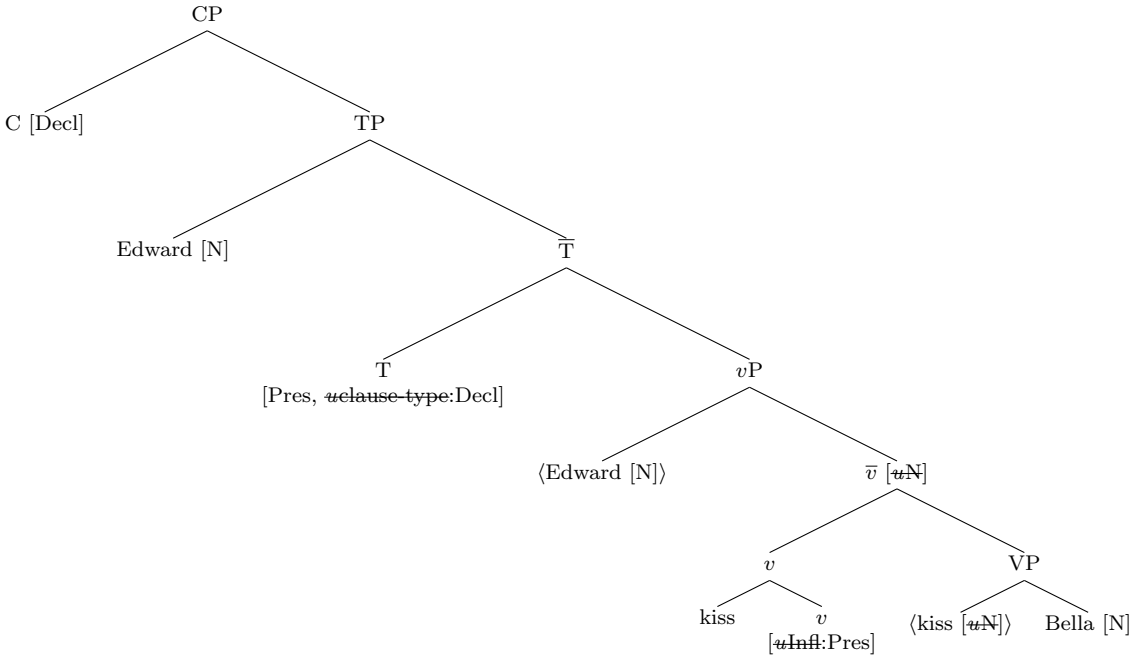


4.3 Summary

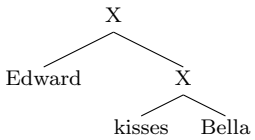
To summarize, we provide an overview of all the principles and technicalities discussed above.

basic unit	syntactic object, tree
syntactic operations	Merge
	Move
	Adjoin
features	valued / privative
	interpretable / uninterpretable
	weak / strong
Full Interpretation	Checking Requirement
	Checking under Sisterhood
	Agree
	Locality of Matching
θ-roles	Unique θ-Generalization
	UTAH
Hierarchy of Projections	C > T > (Neg) > (Perf) > $\cdots$
	$\cdots$ > (Prog) > (Pass) > v > V

Finally, we take a look at the final structure of a simple sentence such as 'Edward kisses Bella'. 'Bella' checks the uninterpretable c-selectional feature on 'kiss'; next, 'kiss' moves to v. The uninterpretable c-selectional feature on v is projected to the bar-level projection, where it is checked by 'Edward'. v also has a [uInfl:]-feature, which can be checked and valued by the [Pres]-feature on T. Since [uInfl:] is weak on v, no moving is necessary. T also bears a [$u$clause-type:]-feature, which can be checked and valued by the [Decl]-feature on C.



In contrast, U-DOP proposes the following tree for this sentence:



Note that this U-DOP representation is very simple and overgenerates widely: as discussed at the end of Section 2.3, U-DOP can generate *all* sentences (but imposes a distribution on them). Nevertheless, as we will show in the next chapters, this simple, overgenerating framework can solve problems of language acquisition; the involved technical machinery as described in this chapter is not necessary.

According to Chomsky et al. (2002), the language-specific component of the brain that is unique for humans, or the Faculty of Language in the Narrow sense (FLN), "comprises only the core computational mechanisms of recursion" (Chomsky et al., 2002, p. 1573).[6] From the overview in the table above, and the exposition in this chapter, it is clear that at least this version of minimalism assumes quite a lot more. Rather, it seems that the U-DOP framework discussed in Part I better fits the image sketched in Chomsky et al. (2002).

[6]However, we believe a good case can be made for recursion being not typically linguistic.

Chapter 5

Subject Auxiliary Inversion

In the previous chapter we have discussed the framework of Minimalist Grammar. In this chapter, and the next two chapters, we will put this to use to account for several linguistic phenomena. Moreover, we will investigate whether the U-DOP approach developed in Part I is also adequate for accounting for these phenomena.

The first phenomenon we will discuss is Subject Auxiliary Inversion (SAI). This phenomenon is one of the most prominent instantiations of poverty-of-stimulus arguments (cf. Crain and Thornton (2006)). The problem lies in the fact that out of (5.1), children could deduce a structure-independent rule like (5.2). However, such a rule would generate structures like the ungrammatical (5.3), rather than the grammatical (5.4). The task of the researcher is thus to explain how children acquire a different rule than (5.2) and produce (5.4) rather than (5.3).

(5.1) the farmer is beating a donkey ⇒ is the farmer beating a donkey?

(5.2) "to form a Yes/No question, move the *first* verbal element is, can, has, ... of the declarative statement to the front" (cf. Crain and Thornton (2006))

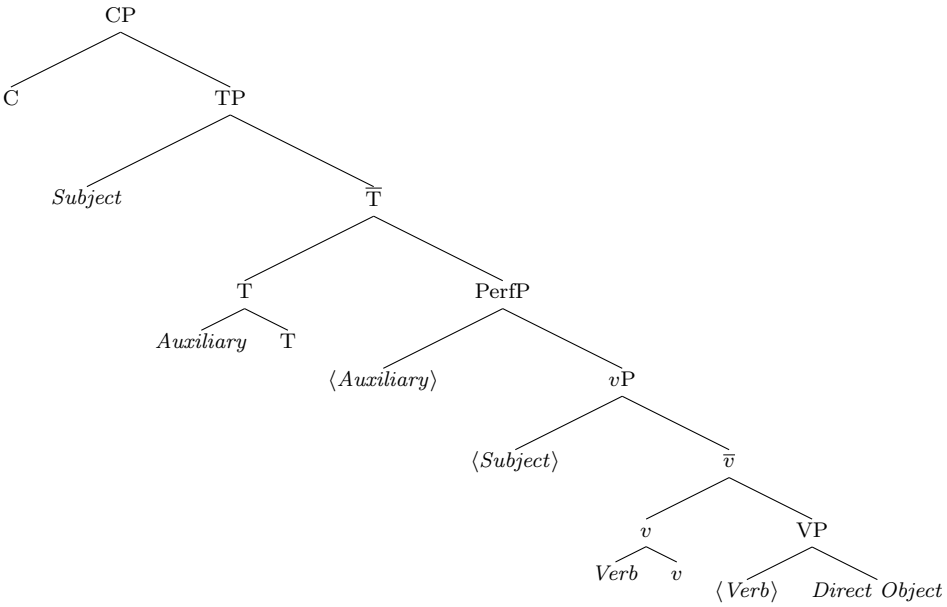
(5.3) the farmer who is beating a donkey is mean ⇒ is the farmer who ⟨is⟩ beating a donkey is mean?

(5.4) the farmer who is beating a donkey is mean ⇒ is the farmer who is beating a donkey ⟨is⟩ mean?

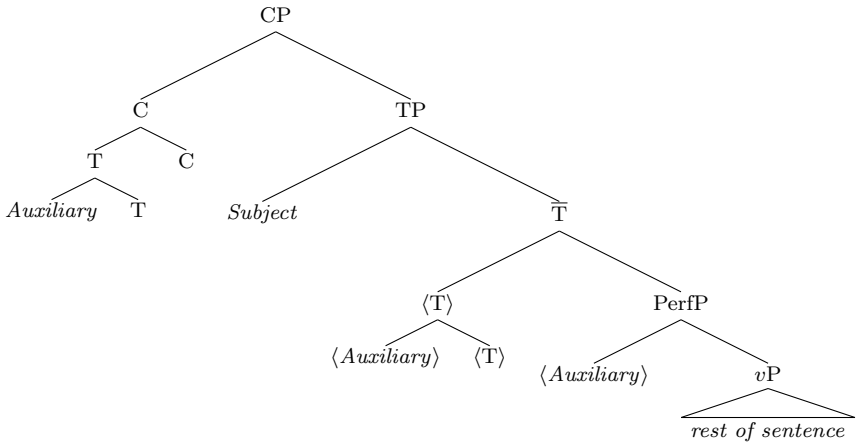
An empirical approach to this problem has already been undertaken in Bod (2009). In this chapter, we will investigate whether our approach can also explain this problem empirically (following the methodology outlined in Subsection 1.2.2). In a first section, however, we will show how the framework of Minimalist Grammar, as described in the previous chapter, deals with Subject Auxiliary Inversion.

5.1 The minimalist account

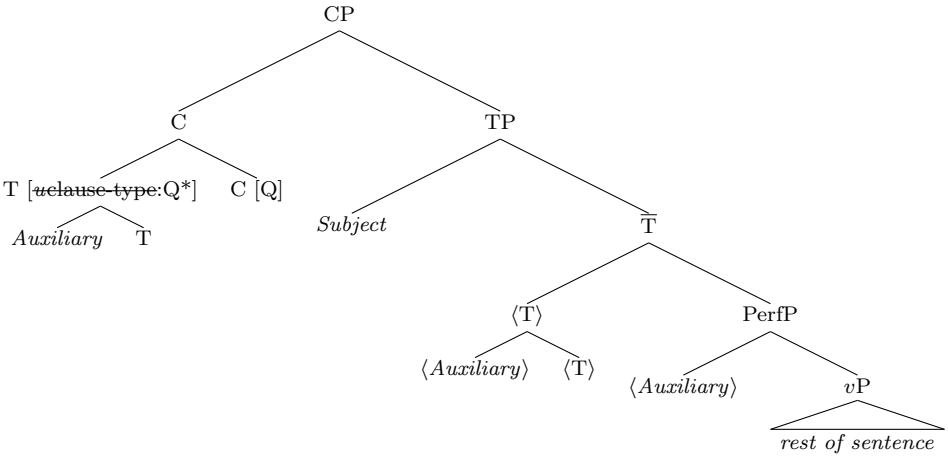
Recall from Section 4.2 the basic structure of a sentence:



It is clear that in yes/no-questions the auxiliary occurs before the subject. Hence, the auxiliary on T must move to a position before the subject, the C position:



This movement is triggered by the [*u*clause-type]-feature on T. This is a strong feature; therefore, T has to move into a local (sisterhood) relation with C, so that the [*u*clause-type]-feature can be valued and checked by the [Q]-feature on C:



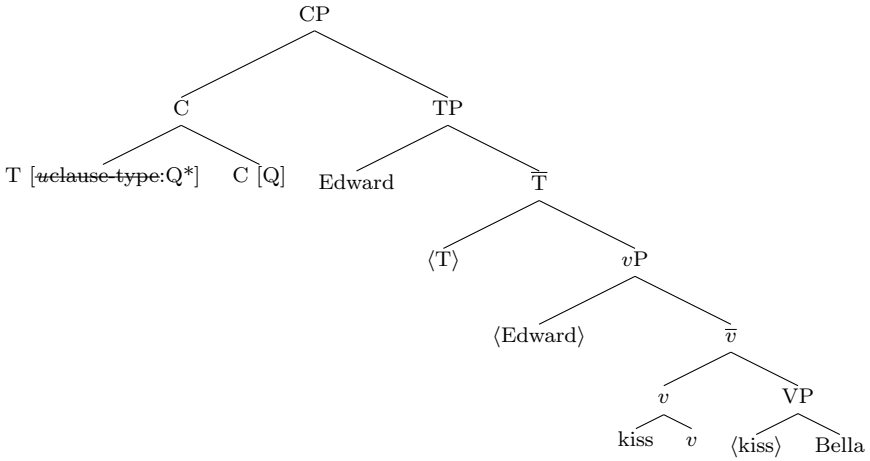
Of course, problems arise when C bears a [Decl]-feature, indicating that the sentence is a statement rather than a question. In that case, this movement should not take place: in that case, the auxiliary occurs after the subject. The solution proposed for this problem is that the [*u*clause-type]-feature is considered strong when valued with [Q], and weak when valued with [Decl]. So when C bears a [Q]-feature, the [*u*clause-type]-feature on T is valued with [Q], and hence considered strong, triggering movement from T to C. When C bears a [Decl]-feature, the [*u*clause-type]-feature on T is valued with [Decl], and hence considered weak, triggering no movement at all.

Finally, we will see how this treatment of yes/no-questions interacts with do-support. For an explanation of do-support, we need a new syntactic entity, the *chain*, and a pronunciation rule, the *Pronouncing Tense Rule* (PTR). The minimalist vision on do-support is that it is a *last resort*: when the PTR cannot apply, do-support will apply instead.

Chain a chain is an object which is formed by an Agree operation. Whenever one feature checks against another, we say that the two syntactic objects entering into the checking relation are in a chain. Each link in the chain must c-command the next one. If c-command does not hold, then the chain is broken. (Adger, 2003, p. 192)

Pronouncing Tense Rule in a chain (T[tense], *v*[*u*Infl: tense]), pronounce the tense features on *v* only if *v* is the head of T's sister. (Adger, 2003, p. 192)

Now consider interrogative questions without an auxiliary (no PerfP or ProgP). This is the structure for the interrogative version of the sentence 'Edward kissed Bella':



Once the movement from T to C has taken place, T no longer c-commands v. Hence, the chain is broken, and the PTR cannot apply. Therefore, we resort to do-support, and we pronounce some form of 'do' on the place of T, and get: 'did Edward kiss Bella?'.

5.2 The U-DOP account

In this section, we will investigate whether an empirical, usage-based account like the U-DOP framework developed in Part I can explain Subject Auxiliary Inversion (following the two-pass model outlined in Subsection 1.2.2). In particular, it has to be able to account for the difference in grammaticality between the following sentences:

(5.5) is the boy eating

(5.6) * is the boy who eating is hungry

(5.7) is the boy who is eating hungry

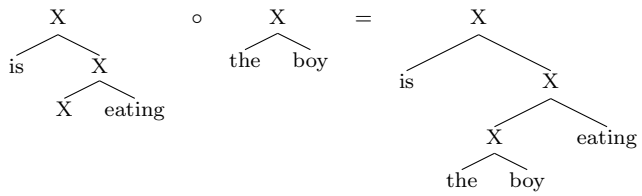
In (5.6), we could say that Rule 5.2 is applied: the first auxiliary is fronted. In contrast, in (5.7), it is not the first auxiliary that is fronted, but the auxiliary from the matrix sentence.

When we try to let our parser (100-best shortest derivations, with ranking as a second phase, cf. Section 3.1) parse these sentences, we run into memory issues: the parsing of (5.6) and (5.7) takes too much memory (more than 30 GB RAM). Therefore, we back off to the second pass of the two-pass model, and look for the shortest derivation ourselves. Now we have to explain why (5.7) is more grammatical than (5.6).

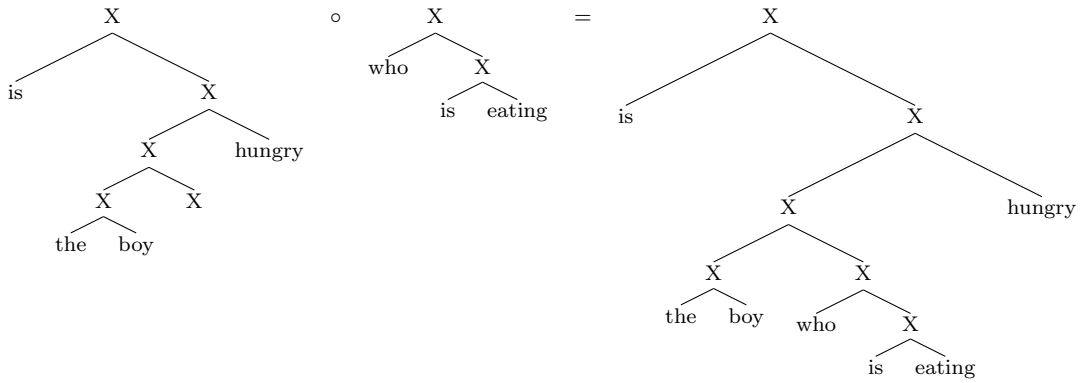
We look at the subtrees from the child-directed speech of the Adam part of the Brown corpus (Brown, 1973), and check how (5.5 – 5.7) can be derived.

First, we investigate whether the entire tree can be found in the subtree-bank: if so, then the shortest derivation consists of that one tree; otherwise, we check whether we can split the tree into two subtrees of the subtree-bank, etc, ...

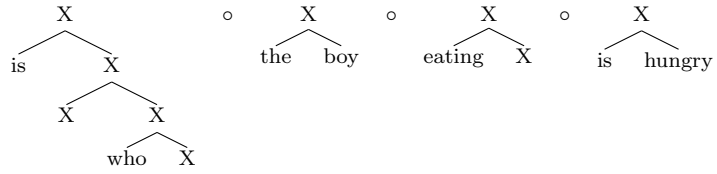
For (5.5), the shortest derivation consists of at least two elements: the entire tree was not found in the subtreebank (ranking: 14,799 + 594 = 15,393):[1]



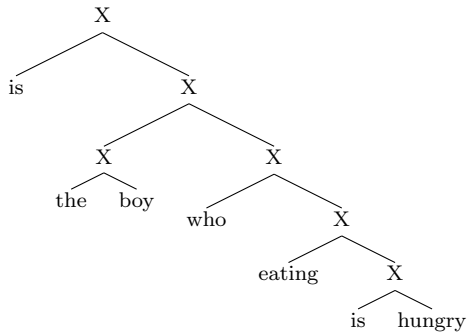
The sentence in (5.7) is of the same complexity: its shortest derivation also consists of two elements (ranking: 1,585,992 + 86,536 = 1,672,528):



Finally, the sentence in (5.6) is the most complex: it only has derivations with four or more elements (ranking: 1,585,992 + 594 + 135 + 4,085 = 1,590,806):



[1] Of course, there are many possible shortest derivations; in our examples, we give just one. We ascertained that there are no shorter derivations, i.e. derivations with fewer elements. Note that we used part-of-speech tags as input to our parser; here we give the words for ease of exposition.

= 

So when a child has to choose between (5.6) and (5.7) to express its thoughts, it will always choose (5.7), because this sentence has the shortest derivation (the ranking does not matter in this case). In this way, the U-DOP framework can account for Subject Auxiliary Inversion: without any rules, the child will always choose the correct alternative, based on the input it has received (in this case, the child-directed speech from the Adam-part of the Brown corpus). So we see that the criterion of shortest derivation works: children will choose that alternative which bears the greatest similarity with what they have already encountered.

In the derivations, we also see that the preference for (5.7) has some linguistic basis: the difference in the derivations is caused by the relative clauses 'who is eating' in (5.7) vs. 'who eating' in (5.6). U-DOP can capture the fact that the first relative clause is more grammatical than the second one, because the latter does not occur in the input.[2] Hence, the derivation of the relative clause-part of (5.6) is more complicated, because we can only use fairly abstract subtrees, whereas for the structure of the first relative clause we have more concrete subtrees at our disposal. We see this for example in the second step of the derivation of (5.7): the subtree (X who (X is X)) can be used, which is a common structure for a relative clause.

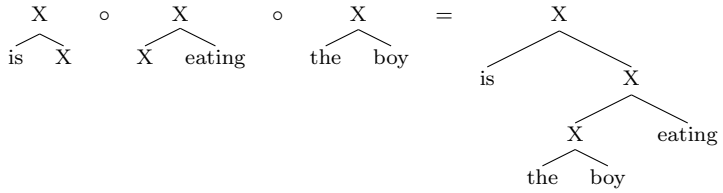
When we look at our other criterion, the ranking, we see that this can also explain SAI. We let the parser developed in Section 3.2 provide analyses for (5.5 – 5.7),[3] and we get the following results.[4] The derivation for (5.5) is (rank: 4 + 16 + 594 = 614):[5]

[2] At least not in the child-directed speech of the Adam-part of the Brown corpus. So it is possible that this relative clause does occur in 'real life' child-directed speech. However, even in that case, it would still be very rare, and hence have little influence.

[3] As was noted in Section 3.2, this parser works more efficient. Hence, we do not run into any memory problems here.

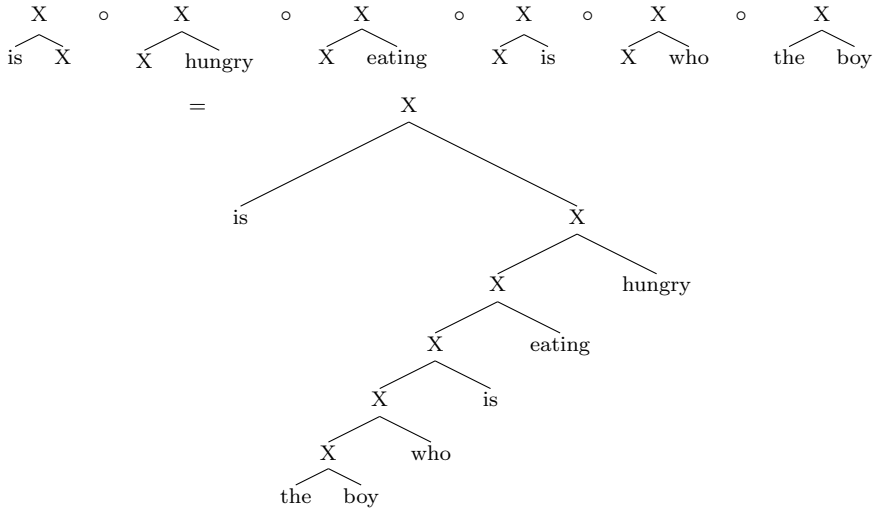
[4] When comparing the ranks of these derivations with the ranks of the derivations above, it is obvious that these derivations will have a much lower rank, since they are meant to optimize the frequency/ranking criterion; however, when comparing the number of elements in the derivations, the derivations above will be the shortest, since they are meant to optimize the maximal overlap/shortest derivation criterion.

[5] Note that these are not assumed to be the trees that would be proposed by a linguist. Since there is no way of knowing what would be the 'correct' trees, i.e. the trees a child would have in mind, we focus on what we can know: the difference in relative grammaticality, formulated in the ranking score. See also the methodological remarks in Subsection 1.2.2 and Section 6.3.

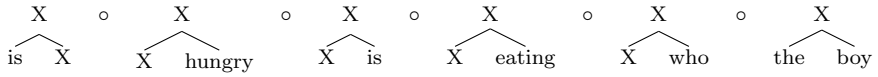


We see that this derivation consists of more elements, but that its composing elements are more frequent (have a lower (and hence, better) rank).

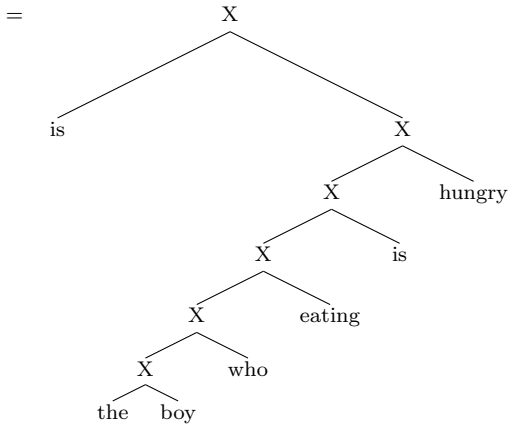
The derivation for (5.7) has a higher rank, since this is a more complex sentence (rank: 4 + 20 + 16 + 13 + 77 + 594 = 724):



Finally, the derivation for (5.6) has the highest rank, so the worst score (rank: 4 + 20 + 25 + 16 + 77 + 594 = 736):[6]



[6]The subtrees of both derivations look the same, and therefore one would expect the rank to be equal; however, the verb 'is' can be assigned two part-of-speech tags, viz. 'v:cop' when it is used as a copula verb, and 'aux' when it is used as an auxiliary. Therefore, the subtrees used in the derivations are not really the same, since different part-of-speech tags for 'is' are used.



So we see that also according to the frequency-based ranking criterion, children will prefer the grammatical (5.7) over the ungrammatical (5.6) when they have to choose between the two alternatives: they will choose that alternative whose composing parts they have encountered most frequently.

5.3 Summary

In this chapter, we have first introduced the minimalist account of Subject Auxiliary Fronting, and shown how this framework can correctly predict differences in grammaticality. However, this account needs the fairly involved machinery of Minimalist Grammar, as described in Chapter 4, and even some additional tweaks have to be made to make everything fit. For example, in the original machinery, features are either strong or weak on a constituent. To account for SAI, however, we must allow that features can be weak on a constituent when they are valued with one value, and strong on that same constituent when valued with another value. Also the analysis of do-support needs new technicalities (e.g. the notion of chains), and a whole new mechanism: a last-resort rule —when all else fails, we apply do-support.

So the minimalist framework can account for SAI, but at a great theoretical cost. The U-DOP framework, in contrast, needs only five things:

1. representations of language expressions: trees
2. fragments of representations: subtrees
3. composition operation: $\circ$
4. disambiguation mechanism: shortest derivation / ranking based on frequency / ...
5. input: a corpus of language expressions that may be assumed to be the input a child has received

Whereas the input plays only a small role in the minimalist framework (in the account of SAI even no role at all —the grammar does all the work), it is crucial in the U-DOP account: both to provide composing elements for derivations (subtrees), as well as to provide information about frequency (rank).

So we see that the poverty of stimulus argument with respect to SAI is refuted. The poverty of stimulus argument says that it is impossible that children learn such complex constructions on the basis of input alone —therefore, innate constraints are necessary. However, in this section we have shown that it *is* possible to account for SAI mainly on the basis of the input, viz. in the simple framework of U-DOP.

Chapter 6

WH-questions

In the remainder of this thesis, we will look at a range of phenomena that in generative linguistics are claimed to be explained by a coherent set of constraints, viz. the *island constraints*. The phenomenon that is most heavily studied in this research area is that of wh-questions. In this chapter, we will investigate this phenomenon in depth; in the next chapter, we will look at related phenomena. First, we go back to the roots of generative grammar: the PhD thesis of Ross (1967), where the island constraints were first developed. Next, we describe how the more recent framework of minimalism, as described in Chapter 4, deals with wh-questions. Finally, we show that it is not necessary to assume innate constraints, but that the U-DOP theory developed in Part I can deal with wh-questions in an empirical way.

6.1 The basic explanation of Ross (1967)

The framework that Ross worked in was that of *transformational grammar*. The syntactic component consists of two parts: a set of context-free phrase structure rules, which generate *deep* structures, and a set of transformations, which convert deep structures into *surface* structures.

WH-questions are formed by the transformational rule **Question**:

(6.1) **Question**

Q	–	X	–	NP	–	Y	
1		2		3		4	$\overset{\text{OBLIG}}{\Longrightarrow}$
1		3 + 2		0		4	

Condition: 3 dominates WH + *some*

In general, such rules state that if you have a configuration like the left-hand-side of the arrow, it obligatorily has to be transformed into the configuration specified in the right-hand-side. Variables like X and Y can range over all strings, including the null string.

So in this case, the configuration we start from has a Q element (a non-pronounced element that signifies we're dealing with a question), then an arbitrary string (labeled with variable X), then a noun phrase, and then again an arbitrary string (labeled with variable Y). The transformation consists in moving the noun phrase to a place between the Q-element and the string X.

For example, the question 'who did Bella kiss yesterday' is derived from 'did Bella kiss who yesterday' as follows:[1]

Q	–	did Bella kiss	–	who	–	yesterday	
1		2		3		4	$\overset{\text{OBLIG}}{\Longrightarrow}$
1		3 + 2		0		4	
Q		who + did Bella kiss				yesterday	

Condition satisfied: 'who' dominates WH + *some*

It is clear that variables like X and Y are indispensable in rules like (6.1): the place of X can be filled by an infinite variety of strings, cf. (6.2 – 6.4) (the string spanned by X is in italics); this infinite variety cannot be captured in a finite conjunction. We say that wh-questions can be infinitely deeply embedded: (6.2) has level 1 of embedding, (6.3) level 2, (6.4) level 3, and in principle we could go on building sentences with ever deeper embeddings. This phenomenon is called 'unbounded scope'.

(6.2) who *did Bella kiss*

(6.3) who *did Jacob say that Bella kissed*

(6.4) who *did Sam order Jacob to say that Bella kissed*

With respect to language acquisition, the puzzle lies in the fact that children only hear constructions of level 1 —but how is it then possible that they can generalize (certainly as adults) this simple construction to more complex ones like (6.3) and (6.4)? Ross answers this question with the variables in Rule 6.1. We will come back to how our U-DOP approach solves this in Subsection 6.3.1.

The immediate next observation, however, is that although variables are indispensable, they also cause problems: by Rule 6.1, we can also derive the ungrammatical sentences (6.5 – 6.8) (again the string spanned by X is in italics).

(6.5) * who *did you know a girl who is jealous of*

(6.6) * what *do you love chicken and*

(6.7) * who *was that you loved* obvious

(6.8) * which *did you read* book

[1] Do-support and auxiliary inversion are not dealt with in Ross' thesis. In the next section, we will discuss the minimalist account of wh-questions, where these details are dealt with.

Now the question is, how do children know that they can generalize from what they hear in (6.2) to (6.3) and (6.4), but not to (6.5 – 6.8)?

To this end, Ross proposes the *island constraints*: these constraints define islands, within which certain rules are constrained to operate.

There is a specific class of rules which obey the constraints. Ross makes a distinction between 'copying' and 'chopping' transformations. The former move a constituent, but leave behind a 'trace' (not in the modern sense of the word, but a pronominal form); the latter move a constituent, and leave nothing behind. Now Ross stipulates that only the latter obey the island constraints. Moreover, only chopping transformations where the chopped constituent is moved over a variable obey the constraints.

In the next section, we will introduce the constraints, and look at their effect on wh-questions. In the next chapter, we will look at other transformations which should obey the constraints, and also at a transformation that doesn't obey the constraints. We will show that, whereas Ross' island constraints only apply to chopping transformations with movement over a variable, our U-DOP approach is more general in that it can capture all kinds of constructions.

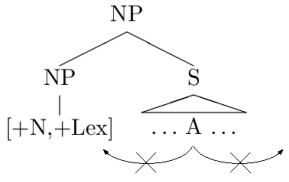
6.1.1 Complex NP Constraint

Ross' first constraint is the Complex NP Constraint. Intuitively, this constraint says: "elements of relative clauses cannot be questioned". Formally, it is stated as follows:

> No element contained in a sentence dominated by a noun phrase with a lexical head noun may be moved out of that noun phrase by a transformation (Ross, 1967, p. 127)

An illustration can be found in Figure 6.1. This means that wh-phrases can

Figure 6.1: An illustration of the Complex NP Constraint



move out of VPs, as in (6.2 – 6.4), but they cannot move out of 'complex' NPs. A complex NP consists of a lexical head (so not a pronominal form like 'it'), and a sentence (most likely, a relative clause). For example, (6.9) and (6.10) are ungrammatical, because 'who' moved out of the complex NP 'a girl who is jealous of ⟨who⟩'.

(6.9) * who did you know a girl who is jealous of?

(6.10) * who did she say you know a girl who is jealous of?

With this constraint, we have now ruled out the first of the ungrammatical sentences in (6.5 – 6.8).

Although there are some counterexamples to this constraint in certain dialects, Ross "believe[s] the Complex NP Consraint to be universal" (Ross, 1967, p. 138). So this constraint is assumed to be part of the language-specific component of the brain that is present in all humans, and hence, to apply in all languages. In Section 6.3, we will see that it is not necessary to assume such a universal, innate constraint, and that this behavior of wh-questions can also be explained on the basis of empirical input alone (together with some general cognitive capabilities).

6.1.2 Coordinate Structure Constraint

The second type of wh-questions which should be constrained has to do with coordination. Consider, for example, (6.11) and (6.12).

(6.11) * what do you love chicken and

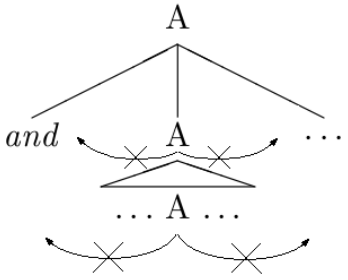
(6.12) * what did Bella read a book and Edward sing

Intuitively, the constraint says: "a conjoined NP cannot be questioned". Formally, Ross formulates this constraint as follows:

> In a coordinate structure, no conjunct may be moved, nor may any element contained in a conjunct be moved out of that conjunct (Ross, 1967, p. 161)

An illustration can be found in Figure 6.2. So (6.11) is ungrammatical, because

Figure 6.2: An illustration of the Coordinate Structure Constraint



'what' is a conjunct in the coordinate structure 'chicken and ⟨what⟩' and hence cannot be moved. Similarly, (6.12) is ungrammatical, because 'what' is an element contained in the conjunct 'Edward sings ⟨what⟩', and according to the constraint, it cannot be moved out of this conjunct.

Now the question how children know that sentences like (6.6) are ungrammatical is answered: also the Coordinate Structure Constraint is considered universal, and part of the language-specific component of the brain.

6.1.3 Sentential Subject Constraint

The third constraint deals with the interesting difference in grammaticality between (6.13) and (6.14):

(6.13) who was it obvious that you loved?

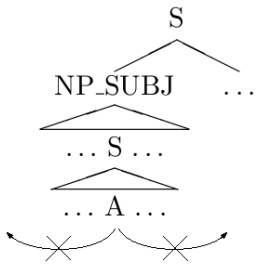
(6.14) * who was that you loved obvious?

Formally, the constraint is formulated as follows:

> No element dominated by an S may be moved out of that S if that node S is dominated by an NP which itself is immediately dominated by S (Ross, 1967, p. 243)

An illustration can be found in Figure 6.3.

Figure 6.3: An illustration of the Sentential Subject Constraint



This constraint applies to structures with a subject NP that itself consists of a sentence (e.g. 'that you loved Edward' in 'that you loved Edward is obvious'). According to the constraint, it is not allowed to move elements out of that NP-sentence. However, when this NP is not the subject of the sentence (e.g. 'that you loved Edward' in 'it is obvious that you loved Edward'), it *is* allowed to move elements out of that NP-sentence. So according to this constraint, (6.13) is grammatical, because 'who' was moved out of the NP-sentence 'that you loved ⟨who⟩' in 'it was obvious that you loved ⟨who⟩'. However, (6.14) is ungrammatical, because 'who' was moved out of the sentence 'that you loved ⟨who⟩' in 'that you loved ⟨who⟩ is obvious'.

We included this constraint for completeness' sake, but it is not really interesting for investigation: sentences of this kind are rarely found in speech, let alone in child-directed or -produced speech. We believe that these structures are learned by schooling, rather than some automatic mechanism.

Interestingly, Ross himself does not consider this constraint to be universal, but rather language-specific. It is claimed that each language has a 'conditions-box'; if a constraint is present in the conditions-box of a certain language, it applies to all relevant rules of that language. So there are languages where the Sentential Subject Constraint does not apply to wh-questions (e.g. Japanese),

but then it also does not apply to other chopping transformations; in languages where the Sentential Subject Constraint applies to one chopping transformation, i.e. it is in the conditions-box, it also applies to all other chopping transformations.

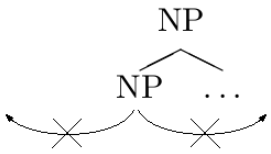
6.1.4 Left Branch Condition

The final constraint Ross proposed to restrict the seemingly unbounded scope of wh-questions is the Left Branch Condition:

> No NP which is the leftmost constituent of a larger NP can be reordered out of this NP by a transformational rule. (Ross, 1967, p. 207)

An illustration can be found in Figure 6.4.

Figure 6.4: An illustration of the Left Branch Condition



This means intuitively that you can only move elements which were right of the head of an NP, not elements left of the head. For example, (6.15) is grammatical, because the entire NP is moved; (6.16), however, is not, because 'which' is moved out of the NP '⟨which⟩ book', where it was the leftmost constituent.

(6.15) which book did you read

(6.16) * which did you read book

Problematically, the condition only speaks of NPs. The difference in grammaticality between (6.17) and (6.18) can then only be explained by analyzing 'how' as deriving from an underlying NP, and stating that the adjective 'sane' is dominated by an NP; otherwise, the Left Branch Condition cannot be enforced.

(6.17) how jealous is Jacob

(6.18) * how is Jacob jealous

Similar to the previous constraint, also the Left Branch Condition is not assumed to be universal, but language-specific. Therefore, it may or may not be present in the conditions-box for a specific language. If it is present, then it applies to all 'relevant' rules, viz. all chopping transformations with movement over a variable.

6.1.5 Summary

In this section, we have seen how the foundations of the generative grammar approach to wh-questions were laid. The seemingly *unbounded scope* was explained by using variables in transformational rules. To rule out ungrammatical sentences which can then arise, certain *constraints* were imposed. These constraints do not only apply to wh-questions, but to all chopping transformations with movement over a variable.

Two of these constraints are assumed to be universal: the Complex NP Constraint and the Coordinate Structure Constraint. The other two constraints, the Sentential Subject Constraint and the Left Branch Condition, are language-specific, and may or may not be present in the *conditions-box* of a language; if they are present, they apply to all relevant rules.

With respect to language acquisition, it is assumed that children have the transformational rule **Question** (cf. (6.1)), and the two universal constraints in the language-specific component of the brain. In this initial version of generative grammar, it is not yet stated how children acquire the (contents of the) conditions-box: how do they know which language-specific rules apply to their language? Later versions of generative grammar claim that this occurs via a system of parameters, which may be set according to the input children receive.

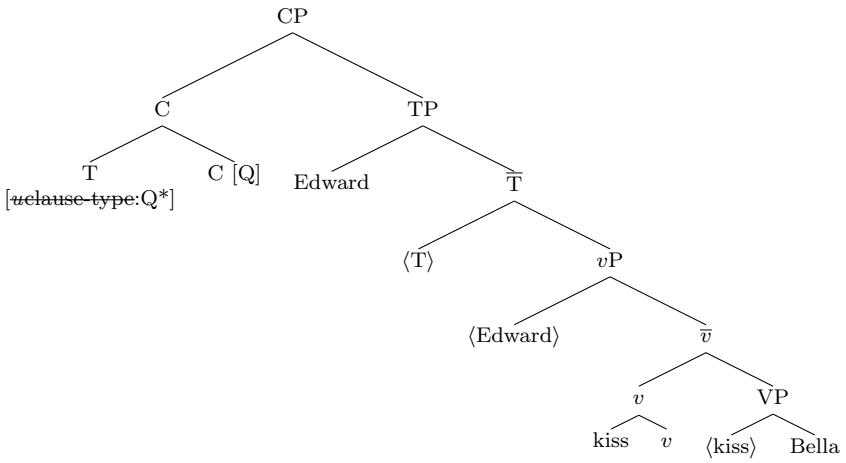
This initial account still leaves many issues on wh-questions open; therefore, we will look in the next section at what the current framework in generative grammar, minimalism, has to say about wh-questions. In Section 6.3 we will then see how our U-DOP approach accounts for the facts concerning wh-questions in an empirical way.

6.2 The minimalist account

The framework of Minimalist Grammar, as developed in Chapter 4, deals with wh-questions in more detail. In this section, we will first discuss the basic explanation for wh-questions. Next, we will focus on several problematic constructions and how they can be solved. Finally, we will discuss the minimalist view on the island constraints of Ross, which were discussed above.

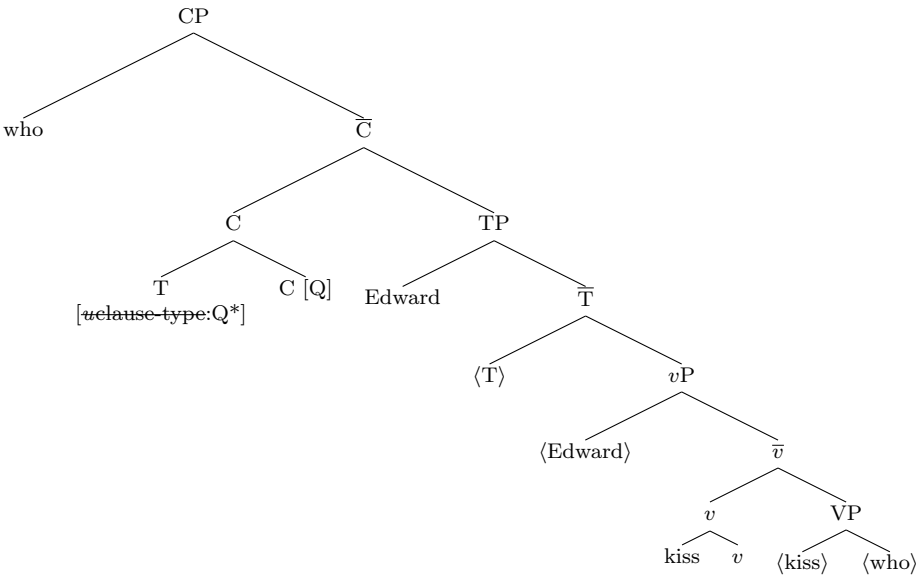
6.2.1 Basic explanation

Recall the basic structure we built in Section 5.1 for a simple yes/no-question, e.g. 'did Edward kiss Bella'. The [*u*clause-type]-feature on T is valued by the [Q]-feature on C, and hence strong. Therefore, T must move into a sisterhood relation with C.

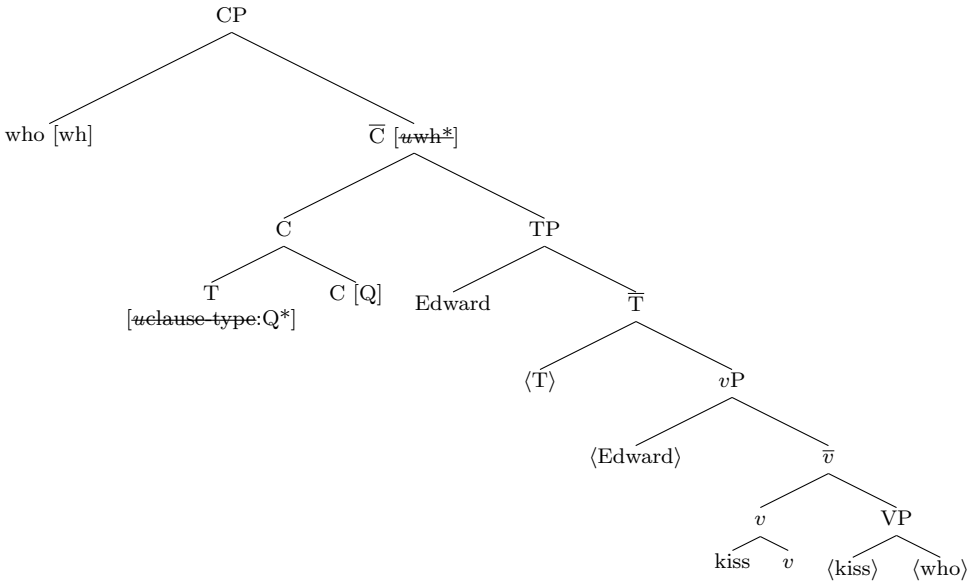


When we now look at a question which contains a wh-element (an element which bears a [wh]-feature), we see that the wh-element occurs before the auxiliary, and hence must be in the specifier of CP. Therefore, it must move to that place from somewhere lower in the structure where it is assigned its θ-role. For example, in (6.19), the wh-element 'who' is assigned its θ-role by 'kiss', and by the UTAH, it must then be a daughter of the VP. To obtain the surface structure, however, 'who' must be moved so that it is in the specifier of CP, before the auxiliary. The structure then looks as follows:

(6.19) who did Edward kiss



Now the question arises what triggers this movement. This is established by proposing an optional, strong [*u*wh*]-feature on C[Q].[2] Since this feature is uninterpretable, it must be checked. It can be checked by the wh-element lower in the structure, but since it is a strong feature, this wh-element must move into a local configuration with C. We can see this at work in the following structure. Note that the [*u*wh*]-feature on C is projected to the bar-level projection.

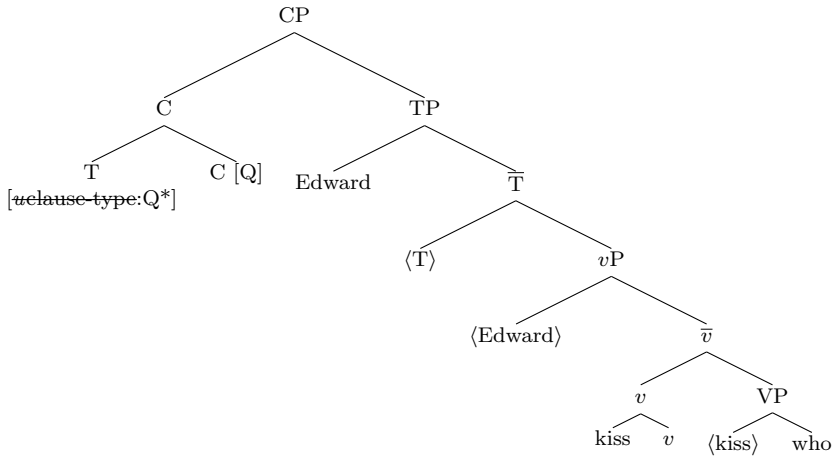


The optionality of the [*u*wh*]-feature on C[Q], however, leads to problems. Sentences where the wh-element is not moved, like (6.20), can be given a grammatical structure: there is no uninterpretable, strong feature that forces the wh-element to move, so it simply stays in place, and all uninterpretable features can be checked.[3]

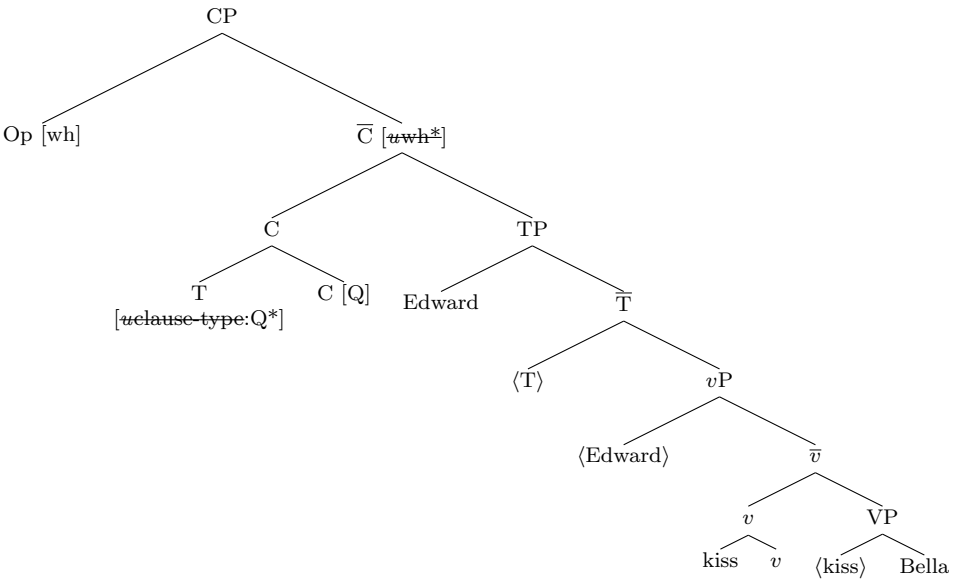
(6.20) * did Edward kiss who

[2]Note that we are now defining features on configurations, rather than on basic categories: the features we introduced earlier either were or were not present on a syntactic category; the [*u*wh*]-feature can only be present on C, when the [Q]-feature is also present.

[3]For an explanation of the appearance of do-support, see Section 5.1.



A possible solution is to make the [*u*wh*]-feature on C[Q] obligatory, rather than optional. In this case, the wh-element must move in order to check the uninterpretable strong [*u*wh*]-feature, causing sentences like (6.20) to be ungrammatical. However, now problems arise with yes/no-questions: they have no wh-element to check the uninterpretable feature; hence, the Checking Requirement (cf. Section 4.1.4) will not be met, and the derivation will eventually crash. The minimalist solution to this problem is to propose a null operator, **Op**, which bears a [wh]-feature. In this way, the [*u*wh*]-feature on C[Q] can be checked by the [wh]-feature on Op. This null operator is Merged directly into the specifier of the CP. Now the structure for 'did Edward kiss Bella' looks as follows:

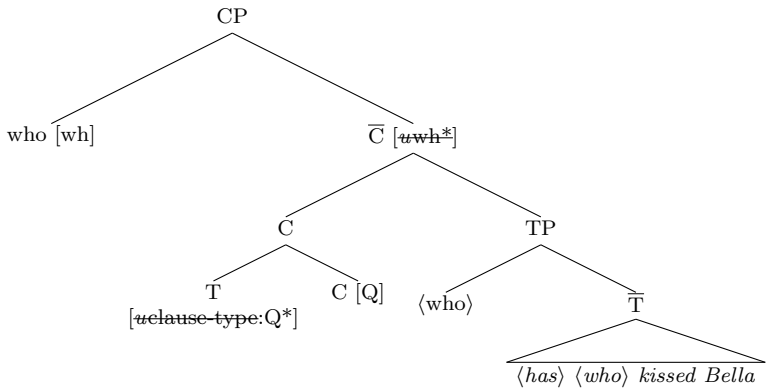


This is the basic set-up for wh-questions in the minimalist framework. However, there are still several problems left to be dealt with. These will be considered in the next subsections.

6.2.2 Subject WH-questions

The first problem that we encounter are subject wh-questions. Naievely, we might propose the following structure for sentences like (6.21). The subject is assigned its θ-role by v; by the UTAH, it originally occurs as the daughter of vP. Then, the [wh]-feature on the subject checks the [$u$wh*]-feature on C[Q], causing it to move into a local configuration with C[Q].

(6.21) who has kissed Bella

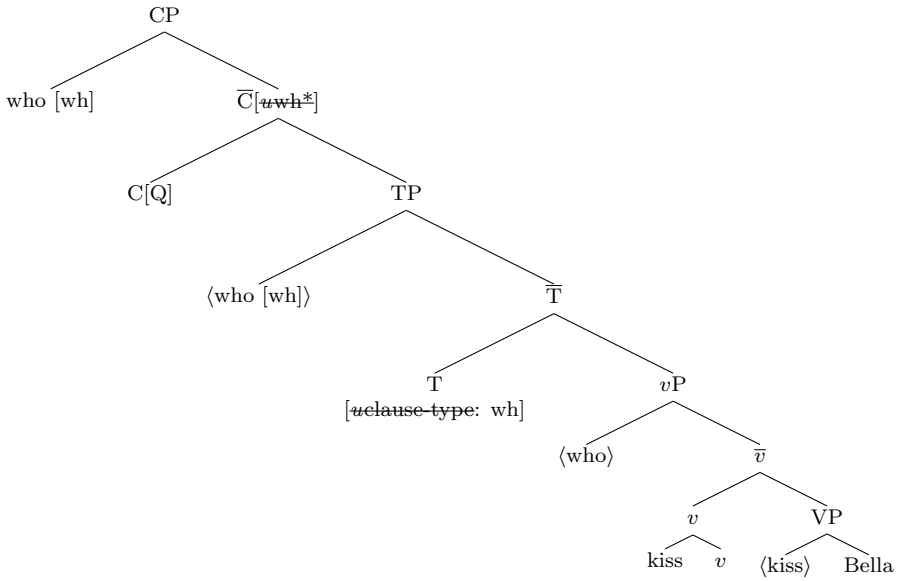


Problematic with this approach, however, is that we still assume T to C movement, triggered by the valuing of [$u$clause-type] on T with the [Q]-feature of C, causing the [$u$clause-type]-feature to be strong, and hence forcing T to move into a local configuration with C. But with sentences like (6.22), where there is no auxiliary, such T to C movement shouldn't occur. If it did occur, the *chain* would be broken (T would no longer c-command v), and do-support would arise (cf. Section 5.1), leading to the ungrammatical (6.23).

(6.22) who kissed Bella

(6.23) * who did kiss Bella

Note that the T to C movement, which is the root of the problem, arises because the [$u$clause-type]-feature on T is valued by the [Q]-feature on C. Therefore, the solution to the problem is to let the [$u$clause-type]-feature on T be valued by something else, viz. the [wh]-feature of the wh-subject. In this way, the uninterpretable feature can be checked, and it is not strong, so no movement will arise. We see this at work in the following structure:



In its original place, the wh-subject allows the checking and valuing of [*u*clause-type] on T; because of this, T needn't be checked by [Q] on C anymore, and hence it doesn't have to move. Next, the wh-subject moves to the specifier of CP to be able to check [*u*wh*] on C. In this way, no uninterpretable features are left unchecked, and the Checking Requirement is satisfied.

6.2.3 WH-questions in situ

The obligatory strong [*u*wh]-feature on C[Q] forces the wh-element to move. However, this is not always desirable: in (6.24) and (6.25), we want the wh-element in italics to stay *in situ*, i.e. in its original place.

(6.24) Edward kissed *who*

(6.25) who kissed *what*

Questions like (6.24) are called *echo questions*, questions like (6.25) *multiple wh-questions*. Minimalist Grammar quickly dispenses with questions of the first type, because they are not considered 'real' questions:

> Echo-questions are usually used to express surprise or amazement, or to simply request that a Part of a sentence should be repeated for clarity. They are not questions in the usual sense of the word, and don't seem to involve the kind of semantics [of regular questions]. (Adger, 2003, p. 352)

Since these are not 'real' questions, it is assumed that the C-element does not bear the [Q]-feature (which marks interrogative statements), and hence it cannot

bear the [*u*wh-feature] (which only appears on configurations C[Q], cf. Footnote 2). However, in the data we are looking at, viz. child-directed speech, echo questions seem to be the preferred way of requesting information. An example from the child-directed speech in the Adam-part of the Childes-corpus (Brown, 1973) is in (6.26). In this example, it is clear that there is no surprise or amazement, and that it is not a request that a part of a sentence be repeated: the mother simply asks for information about 'the little boy's name'.

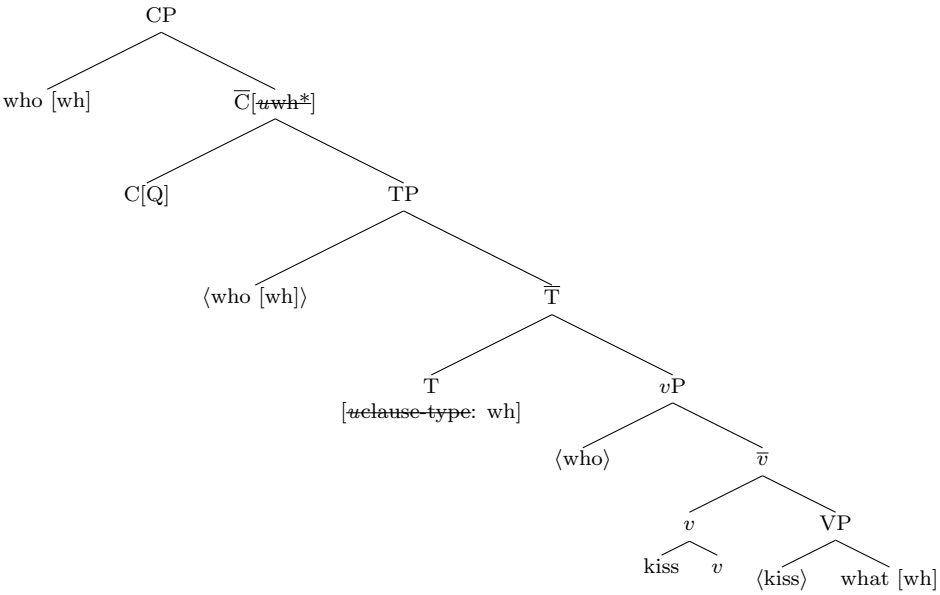
(6.26)

MOTHER:	Rin-tin-tin is the doggie.
CHILD:	no.
MOTHER:	the little boy's name is what?
CHILD:	Rin-tin-tin.
MOTHER:	no, that's the doggie.

(Adam Corpus, file 9, lines 2213 – 2226)

Therefore, we believe that echo questions should be explained with the same mechanism as 'regular' wh-questions. The minimalist framework is unable to establish this; in Section 6.3, we will show how our U-DOP approach is able to do this.

The explanation for multiple wh-questions is fairly straightforward. Building on the analysis of subject questions in the previous section, the following structure for (6.25) is proposed.



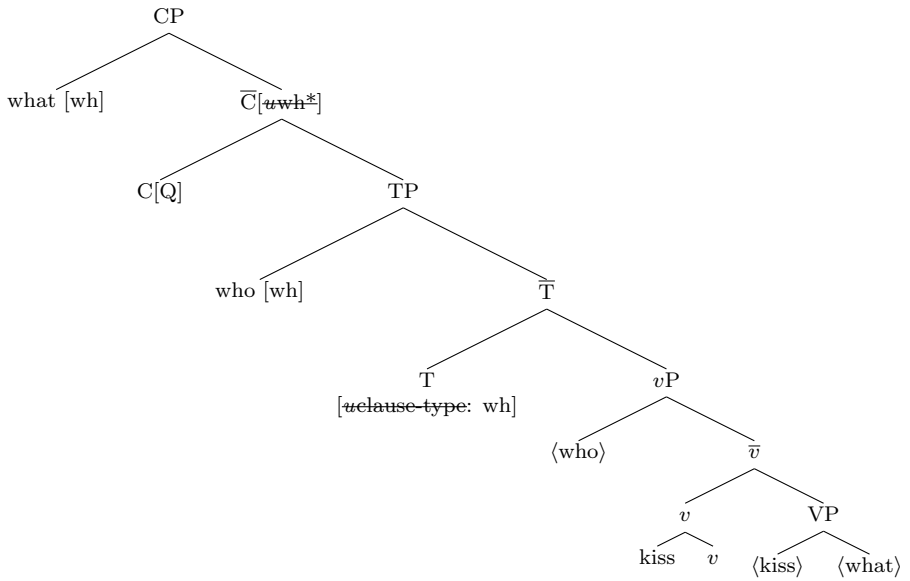
The [wh]-feature on the wh-subject first checks and values the [*u*clause-type]-feature on T, ensuring that no T to C movement (and hence no do-support) takes place. Next, the wh-subject is moved to the specifier of CP where it can check the [*u*wh*]-feature on C. The [wh]-feature of the direct object ('what') doesn't

check anything, but this is not necessary, since it is an interpretable feature. All uninterpretable features are checked, so the derivation is correct.

6.2.4 Superiority

Problematic with the approach in the previous section to multiple wh-questions is that it still has to explain how sentences like (6.27) can be ruled out: isn't it possible that the [wh]-feature on the direct object (rather than the subject) checks the [*u*wh*]-feature on C, and hence moves to the specifier of CP, as in the following structure?

(6.27) * what who kissed



The answer lies in the principle of *Locality of Matching* (cf. Section 4.2.2): a head checks the closest matching feature it c-commands. In (6.27), [*u*wh*] on C cannot be checked by [wh] on 'what', because there is an *intervening* constituent with the feature [wh], viz. 'who'.

Following this line of thinking, it is easy to see that it will always have to be the *highest* wh-phrase that moves; this is called the *Superiority Effect*. We can also see this at work in, for example, (6.28) and (6.29): the direct object is located higher than the prepositional phrase; hence, it is the direct object that moves.

(6.28) who did Bella introduce to whom

(6.29) * who did Bella introduce who to

However, this approach runs into problems with questions like (6.30) and (6.31): in principle (6.30) should be grammatical, and (6.31) not, because the subject is higher than the direct object, and hence it is the subject that should move.

(6.30) which poet wrote which ode

(6.31) which ode did which poet write

In the minimalist framework, this is explained in pragmatic terms, by the D-linking effect. However, we believe that sentences like (6.30) and (6.31) are not fundamentally different from (6.28) and (6.29), and hence should be explained by the same mechanism. In Section 6.3, we will show how our U-DOP approach can establish this.

6.2.5 Embedded WH-questions

When wh-questions are embariseedded, we again run into problems with respect to auxiliary inversion. Since the verbs that select wh-questions can also select yes/no-questions (cf. (6.32) and (6.33)), they have a [*u*Q]-feature, which is satisfied by the Q-feature on C.

(6.32) I wondered if Edward has kissed Bella

(6.33) I wondered who Edward has kissed

Now, similarly as with subject questions, the problem arises that the [Q]-feature on C will value the [*u*clause-type]-feature on T, making it strong, and hence causing T to move to C. But if T moves to C, the auxiliary occurs before the subject, and we would expect the ungrammatical (6.34) and (6.35) to arise, rather than (6.32) and (6.33).

(6.34) * I wondered if has Edward kissed Bella

(6.35) * I wondered who has Edward kissed

The rather ad-hoc solution to this problem is to state that if C is embedded, then it values the [*u*clause-type]-feature of T as weak [Q], rather than strong [Q]. So with respect to the [*u*clause-type]-feature on T, we have the following possibilities:

1. [~~*u*clause-type~~: Decl]: in a declarative sentence, the feature is valued by the [Decl]-feature on C, and it is weak, so no movement arises
2. [~~*u*clause-type~~: Q*]: in a 'regular' interrogative sentence, the feature is valued by the [Q]-feature on C, and it is strong, causing movement from T to C
3. [~~*u*clause-type~~: wh]: in subject questions, the feature is valued, not by a feature on C, but by the [wh]-feature on the wh-subject; the feature is weak, and doesn't have to be checked by C, so no movement arises

4. [~~*u*clause-type~~: Q]: in an embedded question, the feature is valued by the [Q]-feature on C, and it is weak, so no movement arises

6.2.6 Long-distance WH-movement

In the next two subsections, we will see how the minimalist framework is a further development of Ross' initial observations. In this subsection, we will see how long-distance wh-movement, which corresponds to Ross' notion of unbounded scope, discussed in the beginning of Section 6.1, is treated in the minimalist framework. In the next section, we will look at how Ross' island constraints fit in the framework, and at how additional islands can be formulated.

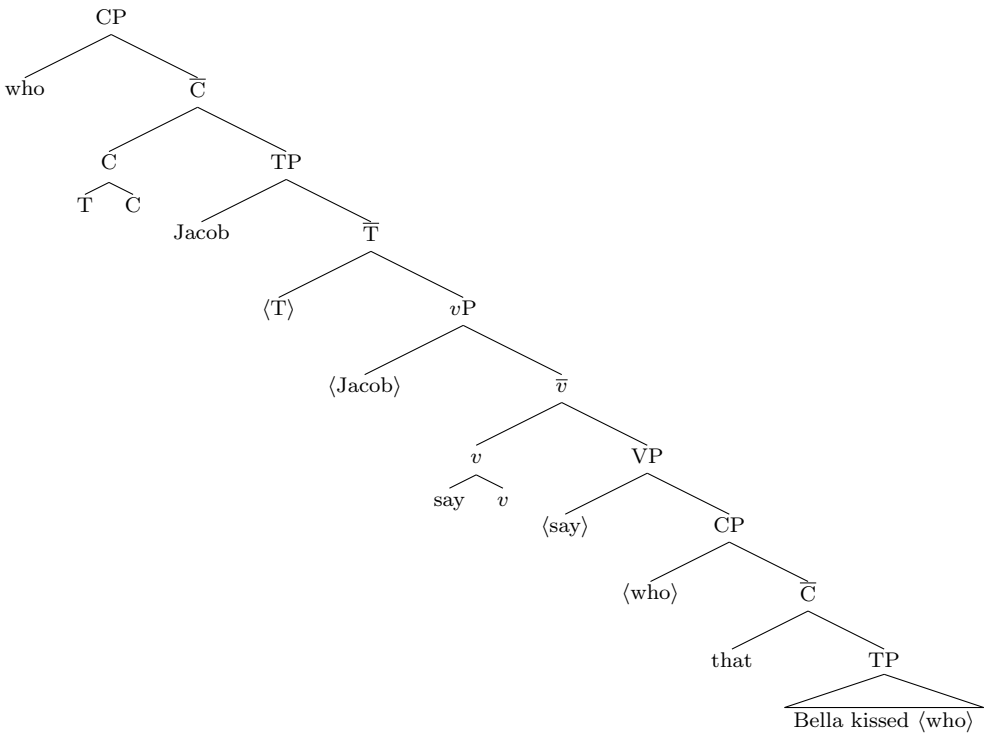
The basic observation is that (except for some restrictions) wh-elements can move to the front of the sentence from an unlimitedly deep position. We repeat the examples given above:

(6.2) who did Bella kiss ⟨who⟩

(6.3) who did Jacob say that Bella kissed ⟨who⟩

(6.4) who did Sam order Jacob to say that Bella kissed ⟨who⟩

It is obvious that the wh-element is assigned its θ-role at its original position (indicated in angled brackets). We also see that the wh-element occurs in front of the sentence in the surface structure. Now the question is: does the movement from the original to the front position occur in one or multiple steps? The minimalist answer is that the movement occurs in multiple steps: "a wh-expression cannot skip a specifier of CP when it moves" (Adger, 2003, p. 363). So a wh-element moves from one specifier of CP to the next. We see this in the following structure for (6.3):



The wh-element 'who' is assigned its θ-role in the lowest TP, by the predicate 'kiss'. Then the first step is movement to the specifier of the lowest CP. In the second step, 'who' moves to the specifier of the top CP, satisfying its [$u$wh*]-feature.

Now the question is: how do we ensure that movement takes place in this way? We define the concept of *phases* (CPs, but also other constituents can be phases) and formulate the *Phase Impenetrability Constraint* (PIC):

Phases constituents which have the property that only their specifier is accessible for feature matching (Adger, 2003, p. 389)

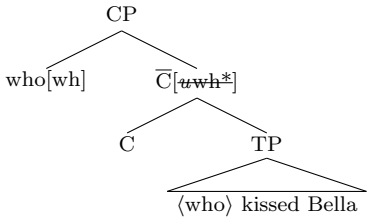
Phase Impenetrability Constraint (PIC) feature matching reaches no further than the specifier of an embedded phase (Adger, 2003, p. 386)

Moreover, we assume that an embedded C can optionally bear [$u$wh*]. Now we can derive long distance wh-questions in a correct way. If we look at the structure for (6.3) given above, we see that, because of the PIC, the wh-element on the bottom of the structure is not allowed to check the [$u$wh*]-feature of the top CP; hence, it cannot move there in one step. However, the wh-element is allowed to check the optional [$u$wh*]-feature on the embedded CP, so it moves to the specifier of the embedded CP. Once the wh-element is in the specifier of the embedded CP, it *is* allowed to check the [$u$wh*]-feature of the top CP, and it can move to its final place, the specifier of the top CP.

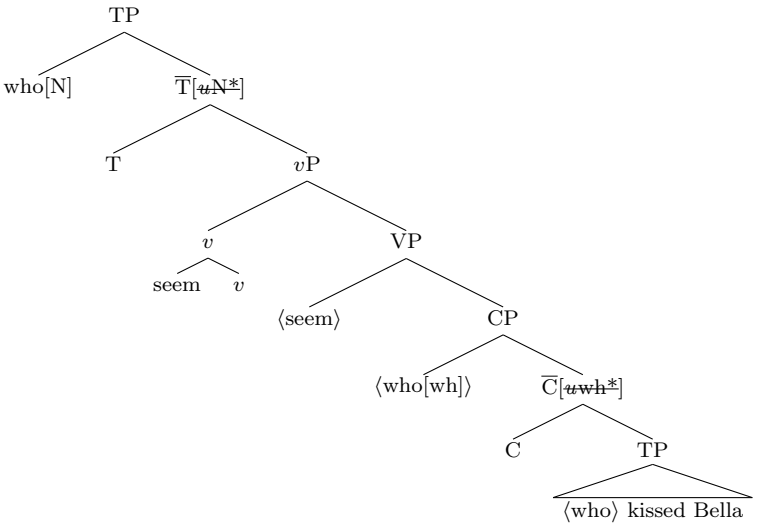
A final problem arises. As the theory stands now, the sentence in (6.36) is considered grammatical:

(6.36) * who seemed kissed Bella

After all, we can have the following derivation. We start with the TP 'who kissed Bella'. We make it into a CP by merging it with C[$u$wh*]. The [$u$wh*]-feature on C can be checked by 'who', which then moves to the specifier of the CP. We have the following intermediate structure:

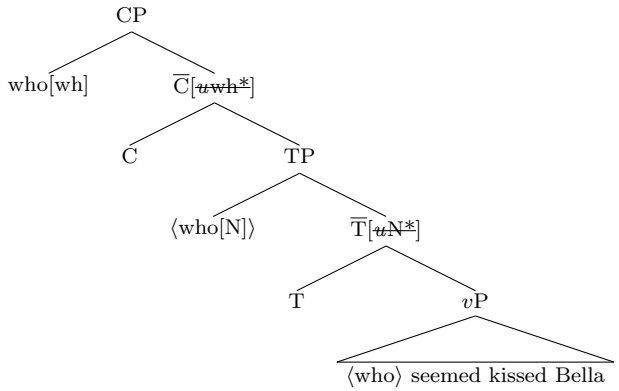


Now we can Merge 'seem', and make it into another TP. Then 'who' moves again to satisfy the [uN*]-feature on T (i.e. to become the subject of the matrix clause).[4]



Finally, we can make this TP into a CP by Merging a C. Then 'who' moves to its final place, the specifier of the top CP, to check the [$u$wh*]-feature on C.

[4] A lot of irrelevant details, such as the discussion of the EPP-feature and the fact that noun phrases are actually determiner phrases, are omitted, because they are not important for the constructions discussed in this thesis.



The (again, rather ad-hoc) solution to this problem is by imposing a restriction on *Improper Movement*:

Improper Movement Restriction only wh-features are visible in the specifier of CP (Adger, 2003, p. 388)

With this restriction, the movement in the second step of 'who' from the specifier of the first CP to the specifier of the TP is not triggered: since 'who' is in the specifier of a CP, only its [wh]-feature is visible; its [N]-feature is not. Hence, it cannot be used to check the [*u*N*]-feature on T, and it does not move to the TP.

Without this step in the derivation, the ungrammatical (6.36) cannot be derived, and the problem is solved.

6.2.7 Islands

In this final subsection, we will see how the minimalist framework deals with Ross' island constraints, and we will look at how additional islands can be defined.

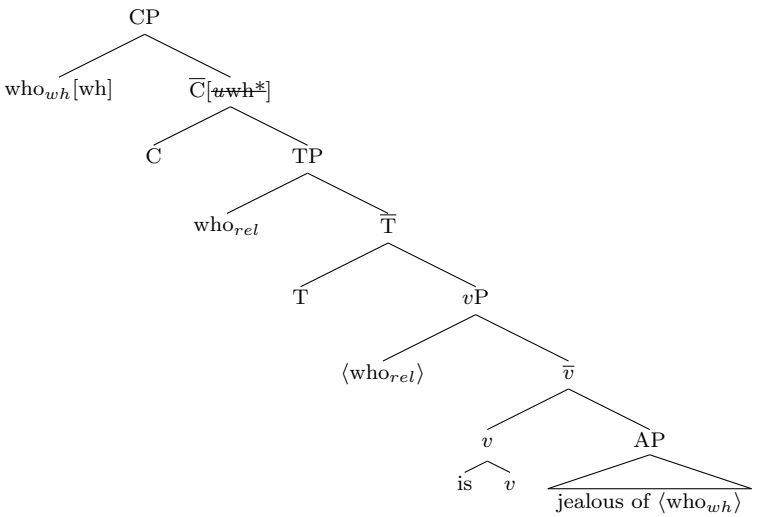
An *island* is defined as "a constituent which is impervious to wh-movement" (Adger, 2003, p. 389). The island constraints as formulated by Ross can be captured by the notion of 'phase' and the PIC.

The Complex NP Constraint (cf. Section 6.1.1) can be captured by treating NPs as phases, like we did with CPs. Informally, the Complex NP Constraint says that elements cannot move out of a 'complex' NP, viz. an NP which itself dominates a sentence (e.g. a relative clause). If NPs are phases, then they have to obey the PIC.

For example, consider (6.9), repeated below. Its ungrammaticality is explained by Ross' Complex NP Constraint, because 'who' is moved out of the complex NP 'a girl who is jealous of ⟨who⟩'. In the minimalist framework, its ungrammaticality is explained as follows. If we want to build a derivation of this sentence, we first build the CP 'who$_{wh}$ who$_{rel}$ is jealous of ⟨who$_{wh}$⟩ '. The

wh-element moves to the specifier of the CP to check the [$u$wh*]-feature of the CP.[5]

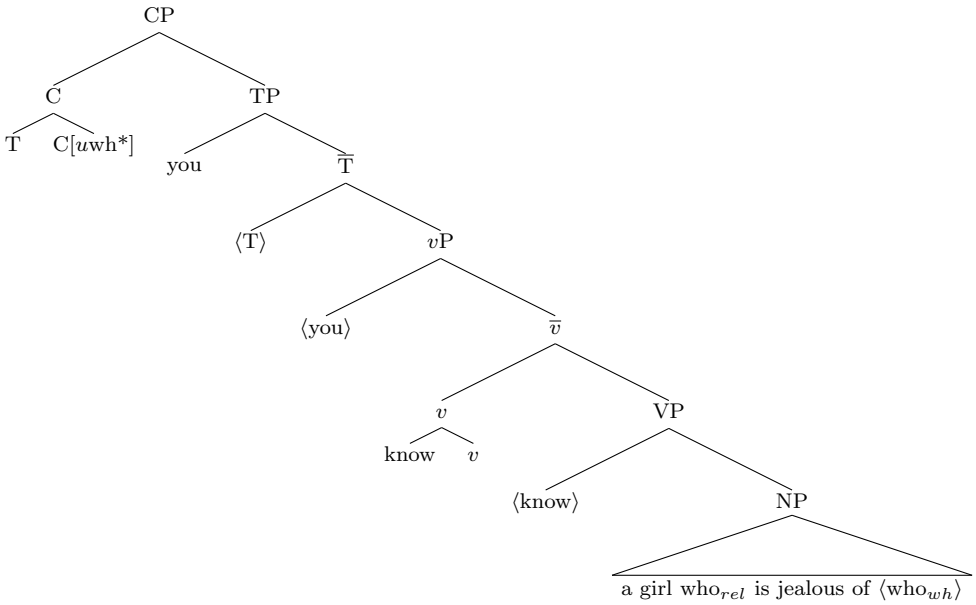
(6.9) * who did you know a girl who is jealous of?



Next, this CP merges with 'a girl' to form an NP, which is then merged with 'know' and the top CP is built.[6]

[5] Note that the subject 'who' is not a wh-element, but a relative pronoun. Hence it cannot be used to satisfy the [$u$wh*]-feature of the CP. To distinguish between the 'wh-who' and the 'relative who', we label them with subscripts: who$_{wh}$ and who$_{rel}$.

[6] Again we omit irrelevant details, like the precise structure of the NP 'a girl who is jealous of ⟨who⟩'.



Now to make the derivation succesful, the uninterpretable [*u*wh*]-feature on C must be checked. The only element that could check this is 'who$_{wh}$'. However, because of the PIC, this is impossible: the NP is a phase, and feature matching cannot reach further than the specifier of an embedded phase, i.e. the [*u*wh*]-feature on C can only be checked by elements up to the specifier of the NP.

Since we cannot build a derivation for (6.9), the sentence is ungrammatical. In this way, Ross' Complex NP Constraint is captured by the more recent minimalist framework. The other constraints developed by Ross (cf. Subsections 6.1.2 – 6.1.4) can be captured in a similar fashion.

Ross' constraints already capture the fact that wh-elements cannot move out of complex NPs, coordinate structures, sentential subjects and the left branch of an NP. In the minimalist framework, we can explain a few additional islands, i.e. other configurations that wh-elements cannot move out of.

For example, also wh-phrases themselves are islands:

> It is impossible to move a wh-phrase out of a CP with another wh-phrase in its specifier. (Adger, 2003, p. 389)

For example, (6.37) is ungrammatical, because 'what' has moved out of the CP 'who saw ⟨what⟩', which has 'who' in its specifier.

(6.37) * what did you ask who saw ⟨what⟩

The ungrammaticality of (6.37) is immediately explained by the PIC and the fact that CPs are phases: the [*u*wh*]-feature on the highest CP cannot be checked by the [wh]-feature on 'what', because 'what' is not in the specifier of the embedded CP —hence, it is unaccessible for feature matching.

6.2.8 Summary

In the last two sections, we have discussed some phenomena which the nativist approach to language can explain: of all given sentences, the theory correctly predicted whether they were grammatical or ungrammatical. However, we had to extend the theory developed in Chapters 4 and 5 a bit: next to the principles and technical apparatus discussed in Sections 4.3 and 5.1, we also needed the following:

null operator	**Op**[wh]
phases	constituents of which only the specifier is accessible for feature matching CPs, NPs, ...
Phase Impenetrability Constraint	feature matching reaches no further than the specifier of an embedded phase
Improper Movement Restriction	only wh-features are visible in specifier of CP
islands	constituents out of which wh-elements cannot move

In the next section, we will investigate whether the much simpler framework of U-DOP, developed in Part I, can also account for these sentences, i.e. whether it can also predict which ones are (un)grammatical.

6.3 The U-DOP account

In this section, we will investigate whether the U-DOP approach developed in Part I can be used to adequately model the acquisition of wh-questions. For most phenomena, the crucial question is: how do children know that sentence A is grammatical, and sentence B is not? Or, in other words, how do children know that they can produce sentence A, but cannot produce sentence B?

In general, the U-DOP answer to these questions is: there is no such thing as absolute grammaticality, there is only *relative* grammaticality. Hence, the distinction between grammatical and ungrammatical sentences is irrelevant; rather, we explain why children *prefer* some sentences over others. So where a nativist approach to language acquisition tries to explain how children know that sentence A is grammatical and sentence B is not, the empiricist approach developed in this thesis tries to explain how it comes about that children prefer sentence A over sentence B.

In this section, we will look at the phenomena discussed in the two previous sections. Our methodology is the following (cf. Subsection 1.2.2). We look at the child-directed speech from the Adam part of the Brown Corpus (Brown, 1973). We assume that this is the input a child receives,[7] so we train our parser on

[7]Note that this assumption makes the problems of language acquisition harder than they are: in reality, a child receives much more input. Therefore, when we can explain phenomena

these sentences.[8] Eventually, we let our parser provide analyses for the sentences under investigation, and we look at the relative scores: the sentence with the best score is considered the more grammatical one.

Finally, we need to make a methodological remark. When we let our parser analyze the sentences under investigation, we only look at the *scores*, we do not look at the *trees*. We believe that it is not relevant whether the tree the parser proposes is identical to the tree a linguist would propose; after all, we cannot know which is the 'correct' tree, i.e. which is the tree constructed by the child. Therefore, we only look at the score of a derivation: we *can* know which sentence should have the best score, viz. the more grammatical one.

6.3.1 Unbounded scope

The first phenomenon we need to account for, is the seemingly unbounded scope that wh-movement has: wh-questions can have infinitely deep levels of embedding. The puzzle lies in the fact that children only hear constructions of level 1, e.g. (6.38) —but how is it then possible that they can generalize (certainly as adults) this simple construction to more complex ones of levels 2 and 3 (e.g. (6.39 – 6.40))?

(6.38) who did you steal from?

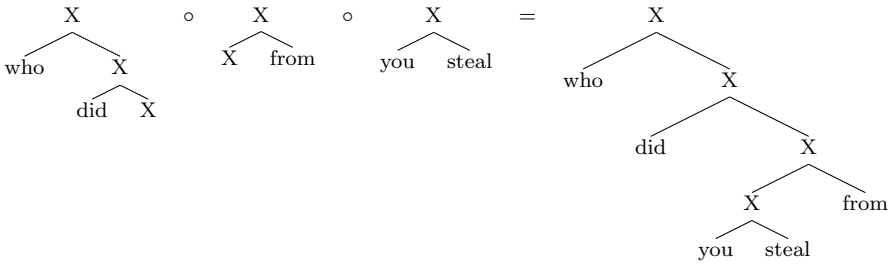
(6.39) who did he say you stole from?

(6.40) who did he want her to say you stole from?

In the previous sections, we saw the initial nativist answer developed by Ross (1967): a transformational rule with variables; in the more recent framework of minimalism, this is explained by a complex interplay between the Phase Impenetrability Constraint and the Feature Checking Requirement.

With our U-DOP theory, the answer is very clear: children can build constructions like (6.39) and (6.40) by simply using fragments they heard before. When we let our parser parse sentence (6.38), we get the following derivation:

(6.41)



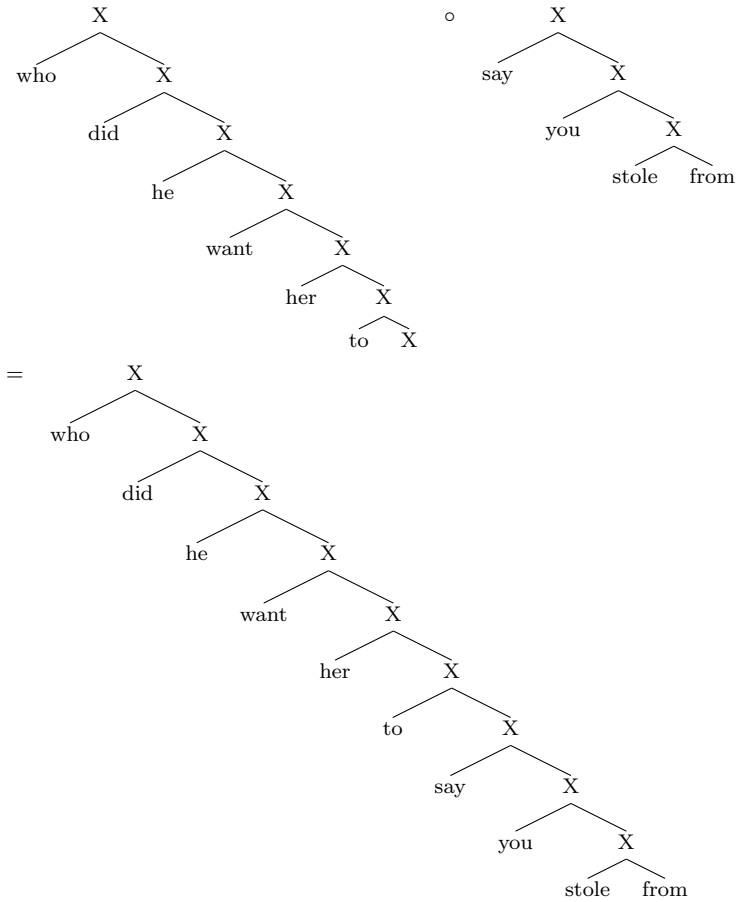
already on the basis of this small input, we can be sure that they can also certainly be explained in reality.

[8] Normally, we will use the parser from Section 3.1, looking at the 100-best shortest derivations, with the ranking as a second phase. Note that we train on the basis of part-of-speech tags; in the examples we will use words for ease of exposition.

Of course, this is not the *shortest* derivation; however, it is the best weigh-off between counting the steps in the derivation, and taking the frequency into account: this derivation consists of three highly frequent subtrees (ranking 1,153 + 7 + 488 = 1,648), whereas the shortest derivation consists of only one subtree (viz. the entire tree) with ranking 26,223.

Sentences (6.39) and (6.40) could not be parsed by the parser, because of memory limitations. Therefore, we back off to the second pass of the two-pass model (cf. Subsection 1.2.2). We find that sentence (6.39) also has the entire tree as its shortest derivation (ranking 2,325,562), and sentence (6.40) can be parsed with as few as 2 subtrees (ranking 1,585,992 and 75,747; totalling 1,661,739), cf. (6.42).

(6.42)



Looking at the speech produced by Adam himself, it is interesting to see that he has only produced (6.38), not (6.39) and (6.40) —so while he *did not* produce these constructions, he *could* have, if he were prompted to do so (under the assumption that children can perform analogical operations).

In this subsection, we have shown how the unbounded scope of wh-questions can be explained in the U-DOP approach, viz. by the combination of fragments

heard earlier. In the next subsections, we will explain how ungrammatical instantiations of wh-constructions can be prevented, without resorting to innate constraints.

6.3.2 Complex NP Constraint

The first problem has to do with 'complex' NPs. We observe a difference in grammaticality between (6.39 – 6.40) and (6.43 – 6.44):

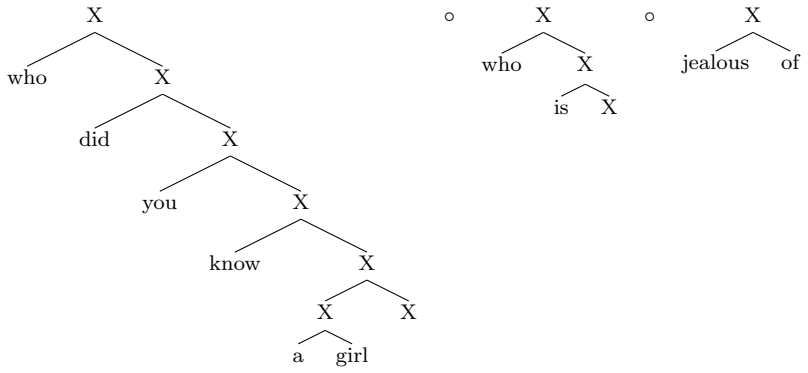
(6.43) * who did you know a girl who is jealous of?

(6.44) * who did she say you know a girl who is jealous of?

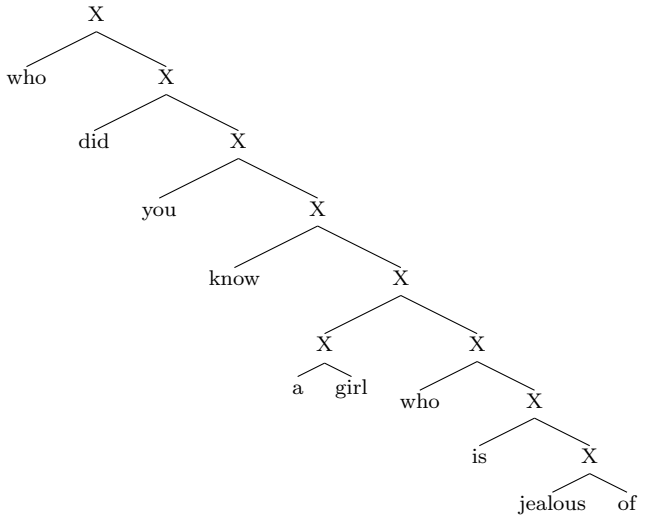
Now the question is, how do children know that they can generalize from what they hear in (6.38) to (6.39) and (6.40), but not to (6.43) and (6.44)? To answer this question, we look at *relative* grammaticality: we compare sentences with the same level of embedding, i.e. (6.39) and (6.43), both of level 2, and (6.40) and (6.44), of level 3.

Similar to (6.39) and (6.40), (6.43) and (6.44) cannot be parsed by the parser, due to memory issues. Therefore, we again backed off to the second pass of the two-pass model. We found that (6.43), unlike (6.39) can only be derived with minimally three subtrees (cf. (6.45)), although its ranking score is better (590,659 + 1,153 + 26,010 = 617,822). On the other hand, (6.40) and (6.44) can both be derived with two subtrees (cf. (6.46)), but in this case, the ranking score of (6.44) is worse (2,325,562 + 1,264,000 + 26,010 = 3,615,572).

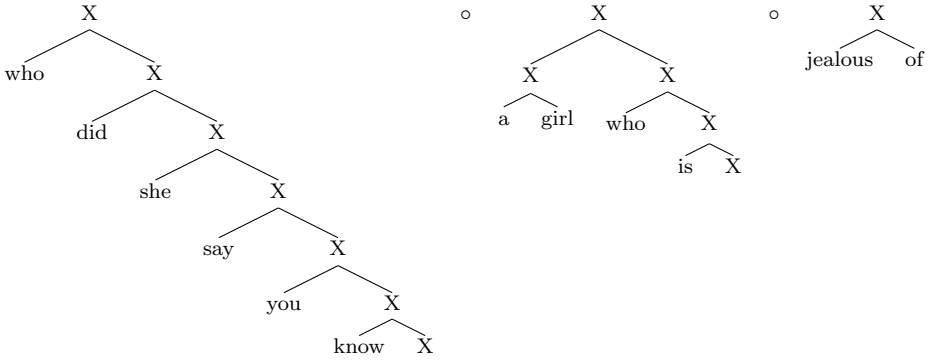
(6.45)

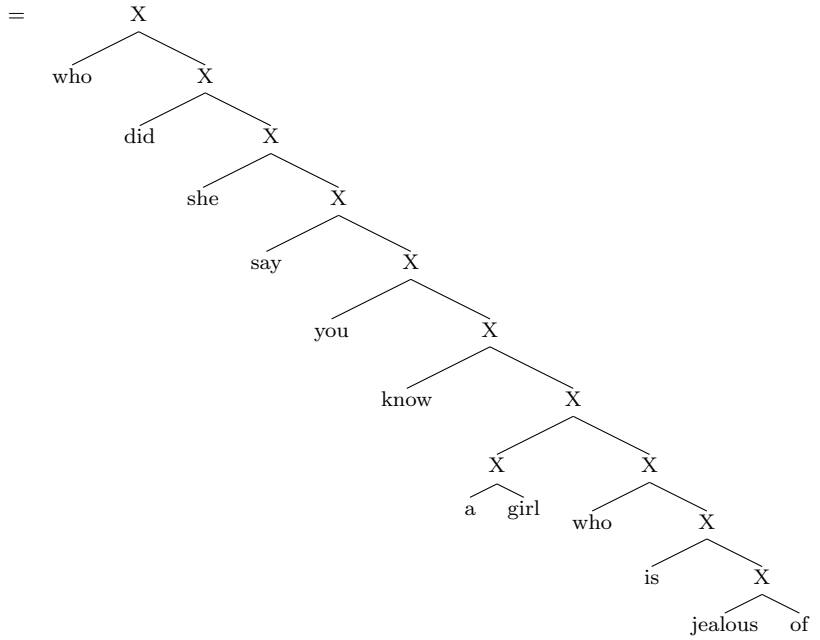


=


X
who
X
did
X
you
X
know
X
X
X
a
girl
who
X
is
X
jealous
of

(6.46)


X
who
X
did
X
she
X
say
X
you
X
know
X
∘
X
X
X
a
girl
who
X
is
X
∘
X
jealous
of



Now we can see that at each level the construction with movement out of VPs is relatively speaking more grammatical than the construction with movement out of NPs. So the Complex NP Constraint emerges naturally from the data: based on the input they receive, children can figure out that constructions as in (6.39) and (6.40) are more grammatical than those in (6.43) and (6.44). The nativist argument that the Complex NP Constraint must be innate, because otherwise children wouldn't be able to tell the difference between generalizing to (6.39) – (6.40) and to (6.43) – (6.44), is now of course directly rebuked: they *can* know the difference according to the U-DOP approach.

A small remark is in order, though. It is remarkable that according to one disambiguation criterion (ranking) (6.40) is indeed preferable over (6.44), but according to another (shortest derivation), we cannot tell the difference in grammaticality. Moreover, according to one criterion (shortest derivation) (6.39) is preferable over (6.43), but according to another (ranking) (6.43) is preferable over (6.39)! We believe this is due to the delicate interplay between desiring on the one hand maximal structural overlap, i.e. the shortest derivation, and on the other hand, the most frequent subtrees, i.e. the lowest ranking score. It is clear that both frequency and derivation length play a role in parsing, but the details of this interplay still need to be further investigated. In any case, it cannot be claimed that there is no non-innate way of telling the difference between (6.39 – 6.40) and (6.43 – 6.44): both can be explained, albeit not yet in a uniform way.

Finally, it is noteworthy that neither (6.43) nor (6.44) are produced by Adam. We could make the same argument here as in Subsection 6.3.1 —that he *could* produce them if he wanted to—, but the parsing results suggest that

if he wanted to express the meaning of (6.43) or (6.44), he would have chosen a 'more grammatical' sentence (with a shorter derivation / better ranking score).

6.3.3 Coordinate Structure Constraint

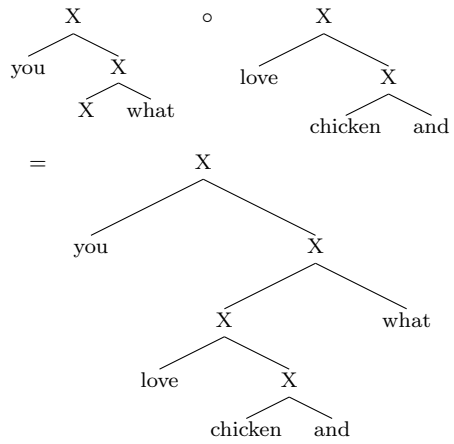
The second problem involves coordination. There is a difference in grammaticality between (6.47) and (6.48).

(6.47) you love chicken and what?

(6.48) * what do you love chicken and?

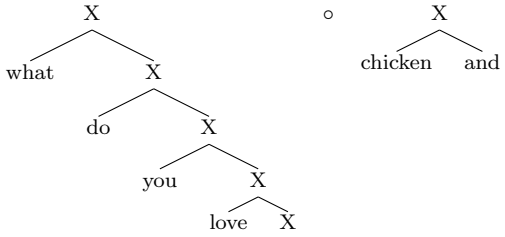
Now the question is: how do children know that (6.48) is ungrammatical? Unfortunately, we cannot develop the same line of reasoning here as above. When we let our parser parse these sentences we get the following results:

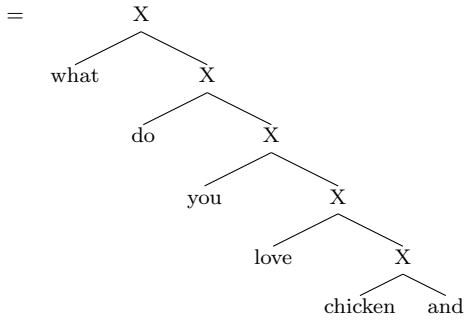
(6.49)



(ranking: 10,595 + 169,145 = 179,740)

(6.50)





(ranking: 5,347 + 9,392 = 14,739)

We see that both derivations are equally short, and that the ranking score of the ungrammatical sentence is even better than that of the grammatical sentence. So within the current framework, we cannot explain how children know that (6.47) is more grammatical than (6.48). However, this does not mean that it is impossible to explain this in an empirical (non-nativist) way.

We think that the reason that the ungrammatical sentence scores so well, is the fact that the subtree (X noun coordinating-conjunction) has an unusually low ranking score. This comes about because the parser takes into account *all* subtrees that can be extracted from the training corpus, also the 'flawed' ones. So every time the parser encounters a sentence containing the sequence "noun coordinating-conjunction" (e.g. in the perfectly grammatical sentence 'Bella loves werewolves and vampires'), the subtree (X noun coordinating-conjunction) is extracted, even though this subtree doesn't occur in the 'correct' parse of the sentence.[9] This inclusion of 'faulty' subtrees is countered effectively by a supervised DOP-parser, since such a parser only looks at the 'correct' trees. Such a parser would then give a much worse score to (6.48), because either it couldn't use (X noun coordinating-conjunction) as a fragment (if it didn't occur), or because (X noun coordinating-conjunction) would have a much worse (i.e. higher) ranking score (because it certainly wouldn't occur very frequently).

However, using a supervised parser would not solve the problem of language acquisition. The question remains: how do children acquire the training corpus of (correctly) parsed trees in the first place? For this problem, we can take the approach developed in Section 3.4. In this approach, an initial corpus (the Extraction Corpus) is trained on in the usual, unsupervised, way. The unsupervised parser resulting from this is used to parse a second corpus (the Held-out Corpus). The result is then a corpus with syntactically annotated trees that can be used as the input for a supervised parser. In this way, we can simulate supervised parsing in an unsupervised way: we still assume no richer input than a raw corpus of part-of-speech tags.

Now the expectation is that subtrees like (X noun coordinating-conjunction) won't occur very often in the parsed HC. Therefore, the (supervised) parser trained on the HC will give a much worse score to (6.48), because either it couldn't use (X noun coordinating-conjunction) as a fragment (if it wasn't used

[9] See Subsection 3.4 for a discussion on the 'correctness' of this analysis.

in the derivations), or because (X noun coordinating-conjunction) would have a lower frequency and hence a much worse ranking score. Unfortunately, applying this methodology to the child-directed speech of the Adam corpus would require parsing half of the corpus (around 11,000 sentences), which the parsers developed in this thesis cannot yet do in a feasible amount of time.

6.3.4 Sentential Subject Constraint

The third constraint applies to constructions with a sentential subject, such as (6.51) and (6.52).

(6.51) who was it obvious that you loved?

(6.52) * who was that you loved obvious?

As noted above, we include them in this overview for completeness' sake, but we do not believe that they are relevant to language acquisition: sentences of this kind are rarely found in speech, let alone in child-directed or -produced speech. We believe that these structures are learned by schooling, rather than by some automatic mechanism.

6.3.5 Left Branch Condition

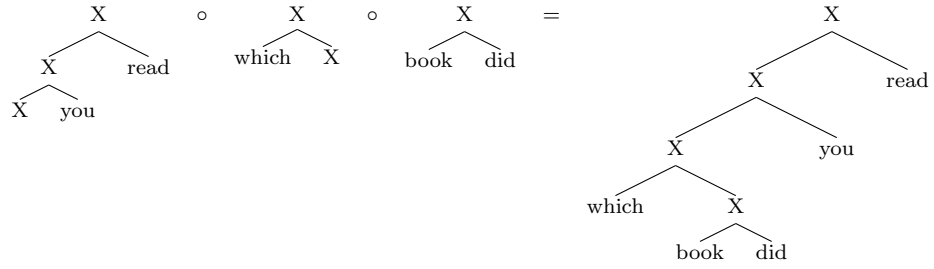
The Left Branch Condition has to do with the difference in grammaticality between (6.53) and (6.54):

(6.53) which book did you read?

(6.54) * which did you read book?

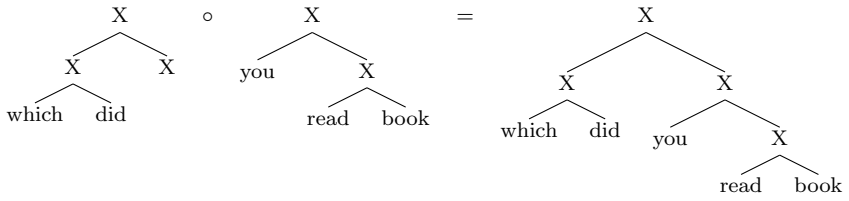
When we let our parser parse these two sentences, we get the following derivations:

(6.55)



(ranking: 608 + 743 + 8,708 = 10,059)

(6.56)



(ranking: 12,809 + 1 = 12,810)

So when looking at the 100-best derivations, we get a derivation of three elements for (6.53), while one of two elements for (6.54). At first sight, one might think that this means that the parser (incorrectly) prefers (6.54) over (6.53). However, this version of the parser also takes the *ranking* into account, and then (6.53) is preferred over (6.54). Moreover, when looking at the shortest derivation only, we find that (6.53) can be parsed in just one step (the entire tree), whereas (6.54) would always need at least two steps. So also with respect to the shortest derivation-criterion (6.53) is to be preferred over (6.54). Hence, it is certainly possible in this case to explain how children can infer that (6.53) is more grammatical than (6.54), without resorting to claims about innate constraints.

Recall that the Left Branch Condition was also invoked to explain the difference in grammaticality between (6.57) and (6.58). However, for this explanation to work the implausible assumption had to be made that 'how jealous' is an NP. We would like for the U-DOP account to explain the difference, without making such an assumption.

(6.57) how jealous is Jacob

(6.58) * how is Jacob jealous

Problematically, however, the parser trained on the Adam-part of the Childes corpus cannot do this. It finds for both sentences a shortest derivation of one tree, because both occur literally in the corpus. Hence, the ranking must be used to decide between both derivations. It turns out that the ranking of (6.57) is much higher (2,325,562) than that of (6.58) (693,021); therefore, the parser would incorrectly prefer (6.58) over (6.57).

A quick inspection of the data reveals why the parser would make such a wrong decision. The reason is that there is only one sentence like (6.57) in the corpus ('how old are you'), and there are five sentences like (6.58); therefore the ranking of (6.58) is better. Of course, it still seems strange that there are five sentences like (6.58), until one sees the examples:

(6.59) why is it dangerous

(6.60) why is it naughty

(6.61) why is it nighttime

(6.62) why is it dark

From this inspection of the data it is clear that the parser only goes wrong because it cannot tell the difference between 'how' and 'why', since it only looks at part-of-speech tags. However, a more advanced version of the parser, which could look at words rather than part-of-speech tags, *would* be able to succesfully recognize the difference, and hence prefer (6.57) over (6.58). So there *is* a way children could learn that from the input; only, we would need a more advanced parser (or a more refined part-of-speech annotation) to actually implement this.

6.3.6 Subject WH-questions

In the next subsections, we turn to issues with wh-questions that are not considered by Ross, but which are explained in the minimalist framework, cf. Section 6.2. We will now explain these phenomena in an empirical way.

The first issue arises with subject wh-questions: we have to explain how children know that (6.63) is the grammatical way of asking such a question, and (6.64 – 6.66) are not.

(6.63) who kissed Bella

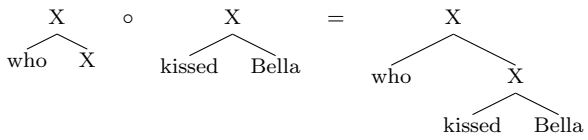
(6.64) * kissed who Bella

(6.65) * did who kiss Bella

(6.66) * who did kiss Bella

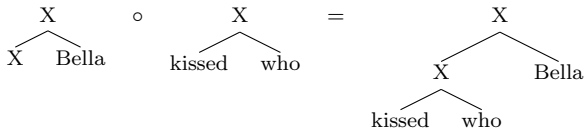
When we let our parser (100-best derivations, with ranking as a second phase) parse these sentences, we get the following derivations:

(6.67)



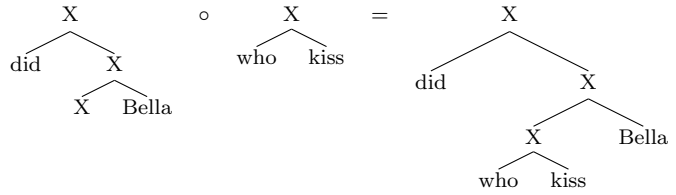
(ranking: 22 + 6,694 = 6,716)

(6.68)



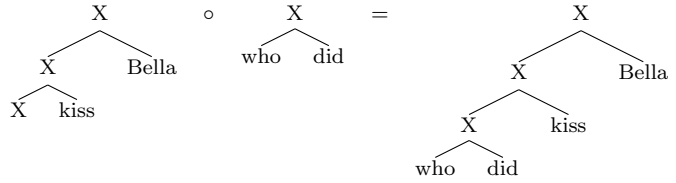
(ranking: 24 + 6,978 = 7,002)

(6.69)



(ranking: 4,230 + 8,527 = 12,757)

(6.70)



(ranking: 4,636 + 2,563 = 7,199)

We see that, although all alternatives have equally short derivations, the parser will prefer the correct alternative, (6.63), because it has the best (= lowest) ranking score. So if we consider the U-DOP parser to be a model of language acquisition, we can explain how children will utter (6.63) when formulating a subject wh-question, and not (6.64 – 6.66): the former is preferred over the latter.

So we see that also the phenomenon of subject wh-questions can be explained without resorting to the complex, supposedly innate machinery of minimalism; in this subsection we have shown that it is at least possible to predict the (most) grammatical alternative, solely based on the input children receive and the simple U-DOP framework.

6.3.7 WH-questions in situ

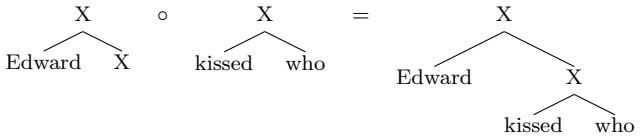
Recall from Subsection 6.2.3 that there are two types of wh-questions in situ: echo-questions, such as (6.71), and multiple wh-questions, such as (6.72). The former are not really explained in the minimalist framework: they are not considered 'real questions'; the latter are explained via the regular processes of feature valuing and checking.

(6.71) Edward kissed who

(6.72) who kissed what

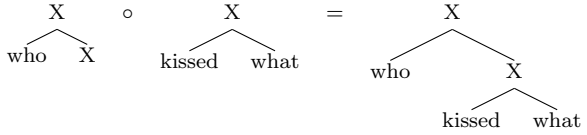
In the U-DOP framework, these constructions are fairly easily generated. We let our parser parse them, and get the following derivations:

(6.73)



(ranking: 21 + 6,978 = 6,999)

(6.74)

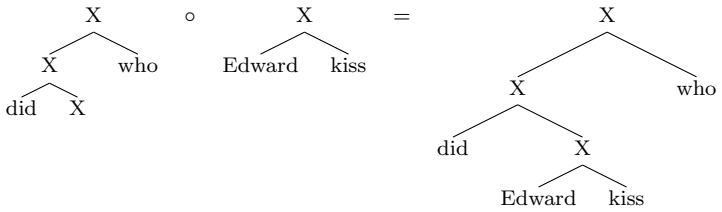


(ranking: 22 + 6,978 = 7,000)

For example, an alternative for (6.71) such as (6.75) is considered less grammatical: its derivation has a much worse ranking score.

(6.75) * did Edward kiss who

(6.76)



(ranking: 11,282 + 10,109 = 21,391)

So the U-DOP framework provides a very simple explanation of wh-questions in situ, without considering them something special, or developing special technical tricks: children can simply generate these type of questions using fragments they heard before.

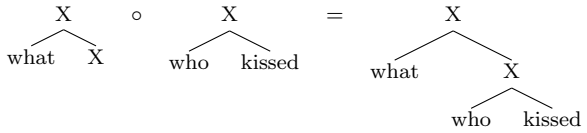
6.3.8 Superiority

In Subsection 6.2.4, we observed that a *superiority effect* takes place with multiple wh-questions: the 'superior' wh-phrase is the one that gets moved. However, in certain cases this effect does not occur. Therefore, the minimalist account had to take refuge in a combination of syntax and pragmatics to deal with all cases of superiority. In this subsection, we will show how the U-DOP account can deal with all cases in a uniform matter.

In the first place, we need to explain why multiple wh-questions are formed like (6.72) from the previous subsection, and not like (6.77). The derivation by our parser of (6.72) can be found above in (6.74); the derivation of (6.77) is in (6.78). We clearly see that, although both derivations are equally long, the parser will prefer (6.74), because its ranking score is better.

(6.77) * what who kissed

(6.78)



(ranking: 22 + 8,527 = 8,549)

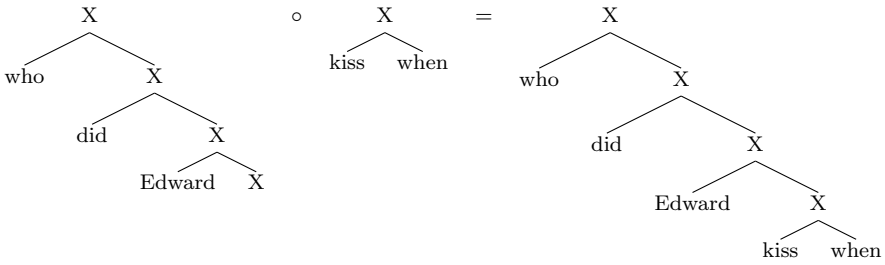
Second, we see that the superiority effect also plays with other types of constituents. For example, (6.79) is grammatical, but (6.80) is not, because in the original structure 'Edward kissed ⟨who⟩ ⟨when⟩', 'who' is the superior wh-phrase, and hence needs to move (according to the minimalist framework).

(6.79) who did Edward kiss when

(6.80) * when did Edward kiss who

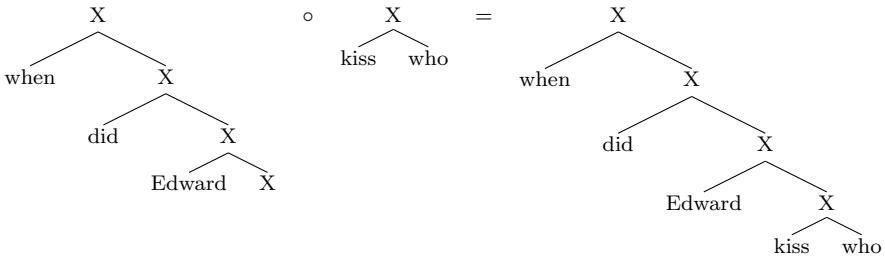
Our parser could not parse these sentences because of memory issues. Therefore, we backed off to the second pass of the two-pass model and checked manually what the shortest derivations are (cf. Subsection 1.2.2). The derivation for (6.79) is in (6.81), that for (6.80) in (6.82).

(6.81)



(ranking: 247,312 + 21,404 = 268,716)

(6.82)



(ranking: 287,863 + 6,978 = 294,841)

Again the U-DOP theory makes the correct predictions: (6.79) is preferable, because its derivation has a lower ranking score.

Finally, we need to account for cases where the superiority effect does not play. For example, both (6.83) and (6.84) are grammatical. The minimalist framework cannot in itself explain this; it needs to resort to pragmatics.

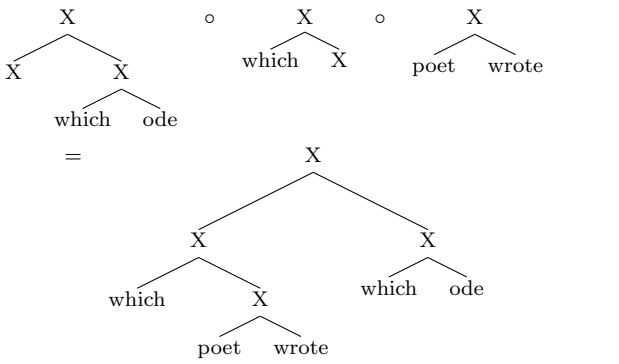
(6.83) which poet wrote which ode

(6.84) which ode did which poet write

In U-DOP, such issues do not arise. As we discussed in the end of Section 2.3, U-DOP generates *all* sentences. So in principle, all sentences are grammatical. This does not mean that all sentences are equally likely to be uttered: U-DOP imposes a *distribution* on the set of all sentences: some sentences are more grammatical, i.e. more likely to be uttered, than others.

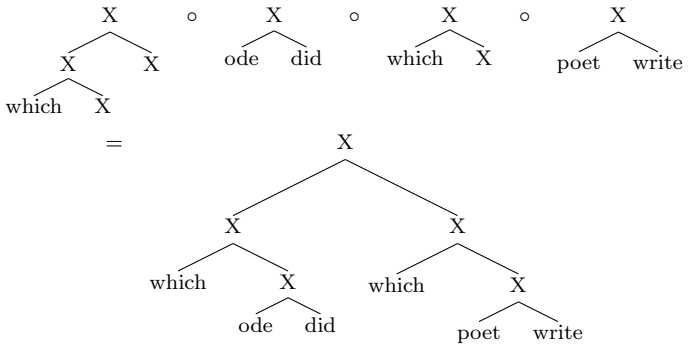
The fact that both (6.83) and (6.84) can be generated is therefore trivially explained: since all sentences can be generated, also these sentences can be generated. (The U-DOP approach has no problem to explain why some sentences can be generated; it has more difficulty to explain why some sentences *cannot* be generated, cf. Footnote 4 of Chapter 7.) When we let our parser parse both sentences, we get the derivations in (6.85) and (6.86).

(6.85)



(ranking: 29,081 + 743 + 9,060 = 38,884)

(6.86)



(ranking: 8,274 + 8,708 + 743 + 9,060 = 26,785)

We see that (6.85) is preferred over (6.86) when taking the shortest derivation as the criterion, but (6.86) is preferred over (6.85) when taking the ranking into account. In any case, we see a different pattern here from the examples with superiority above. Hence, we have eluded the superiority effect for these special cases, without having to do anything special: we can account for both the cases with superiority (6.77 – 6.80) and those without (6.83 – 6.84) in a uniform way.

6.3.9 Embedded WH-questions

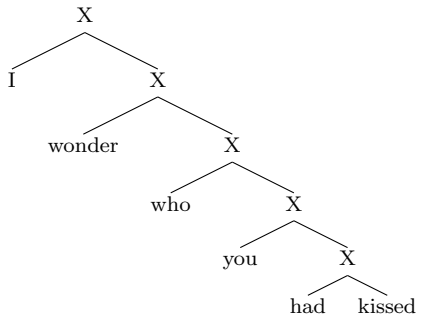
Another problem which the minimalist framework can only solve with an ad-hoc solution, has to do with embedded questions. With embedded questions, auxiliary fronting mustn't occur. Hence, (6.87) should be considered grammatical and (6.88) not.

(6.87) I wonder who you had kissed

(6.88) * I wonder who had you kissed

In the U-DOP framework, this can be explained fairly straightforward. We could not let the parser parse the sentences, due to memory issues, so we did a manual check of possible derivations. Both sentences can be derived in one step, cf. (6.89) and (6.90), so the shortest derivation criterion cannot be used for disambiguation. However, when we look at the ranking scores, we see that the U-DOP approach correctly prefers (6.89).[10]

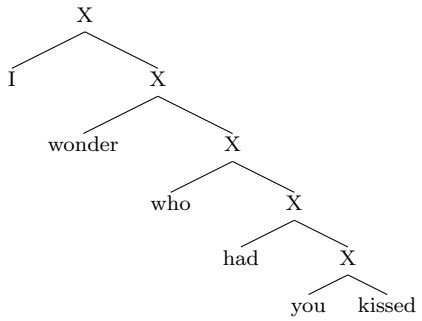
(6.89)



(ranking: 1,585,992)

[10]It might seem surprising that trees like (6.90) are found at all in the corpus; however, this is explained when we look at the actual sentences from which this tree is derived: 'what do you mean what do I want' and 'let's see what shall we make'. These are not 'real' embedded sentences, with indirect speech, but the first sentence uses *reported* speech, and the second sentence merely puts two questions next to each other. When we use punctuation, this is immediately clear: 'what do you mean: "what do I want"' and 'let's see, what shall we make'.

(6.90)



(ranking: 2,325,562)

So we see that the U-DOP framework can also account for this construction in a simple and straightforward way, in contrast with the minimalist framework, which needs technical tricks to accomplish this.

6.3.10 WH-islands

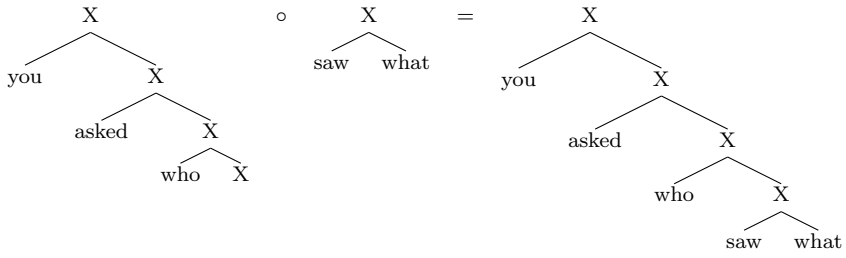
Finally, we saw that the minimalist framework was also capable of dealing with 'other islands', i.e. other constituents that wh-phrases cannot move out of. An example of such an island are wh-islands: wh-phrases cannot move out of wh-phrases. We see this in (6.91) and (6.92): the former is grammatical, but the latter is not, because 'what' has moved out of the wh-phrase 'who saw ⟨what⟩'.

(6.91) you asked who saw what

(6.92) * what did you ask who saw

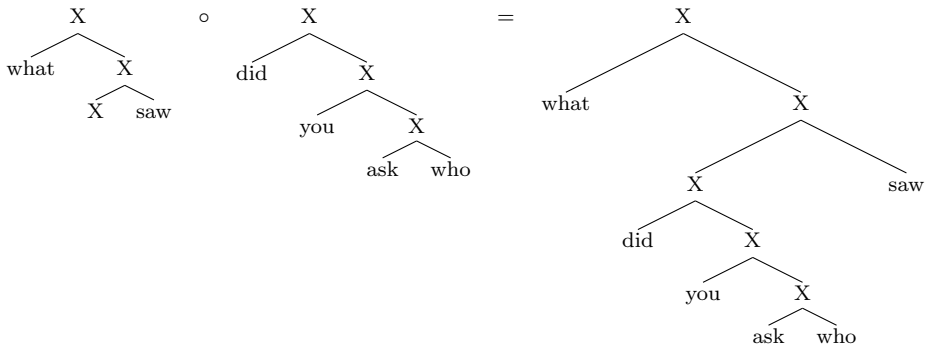
The U-DOP framework can also account for these sentences easily. We let our parser parse both sentences, and we find the derivations in (6.93) and (6.94).

(6.93)



(ranking: 14,231 + 6,978 = 21,209)

(6.94)



(ranking: 2,892 + 35,439 = 38,331)

Both alternatives have a shortest derivation of two steps, but we can use the ranking as a disambiguation criterion: (6.93) has a lower ranking than (6.94) and is hence preferable. Hence, U-DOP makes the correct prediction that (6.91) is the more grammatical sentence.

So the additional islands that the minimalist framework proposes can also be explained by U-DOP in a very straightforward way.

6.4 Conclusion

In this chapter we have studied the phenomenon of wh-questions in depth. First, we looked at the basis for every nativist account of wh-questions, viz. the PhD thesis of Ross (1967). This account already captured basic facts concerning the construction, but still left open many details. Next, we discussed the account of wh-questions in a more recent nativist framework, viz. minimalism, building further on the theory described in Chapter 4. Finally, we showed that each of the problems that occurred in the discussion of the two nativist frameworks could be accounted for by the usage-based U-DOP theory developed in Part I. We did not constrain ourselves to only showing that the theory could *in principle* explain the phenomena, but we conducted *actual* computational experiments, using the implementations from Chapter 3. With this methodology, we could model the acquisition of wh-questions on the basis of (a small portion of) the real-life data that a child receives —this makes the U-DOP approach all the more plausible.

Interestingly, the main difference between the nativist and the U-DOP account of wh-questions is the role of the input. In the nativist account, the construction is mostly explained by involved theoretical machinery; in the U-DOP account, however, the machinery is fairly limited, and the input plays a large role.

Note that we do not claim to have given *the* correct way of modeling the acquisition of wh-questions. Rather, we hope to have proven false the nativist argument that these constructions cannot be explained on the basis of input

alone, by showing that it is at least *possible* to explain these constructions with (a version of) the usage-based U-DOP theory.

Chapter 7

Related phenomena

Recall from Section 6.1 that Ross claimed that the constraints he proposed to account for wh-questions are fairly general and apply to a well-defined class of other rules as well. In this section, we will show that our U-DOP account of Part I is at least equally general, by showing that it also can account for these other phenomena. Moreover, we will show that the U-DOP account is even more general, by applying it to a phenomenon that is no element of the class of rules obeying the constraints, viz. left dislocation.

Since the examples for the phenomena in this chapter are fairly intricate, we will find that more often than not the implementation of the parser is not able to analyze the sentences due to memory issues. However, we believe this reflects the way children would respond to such sentences: also they would have difficulty with processing them. Therefore, also with respect to processing difficulties, the U-DOP approach is more adequate at modeling language acquisition than the nativist account.

7.1 Relative clause formation

The phenomenon that is most closely related to wh-questions is relative clause formation. It, too, is accounted for by a transformational rule:

(7.1) **Relative Clause Formation**

W	–	$[_{NP}$	NP	–	$[_{S}$	X	–	NP	–	Y	$]_S]_{NP}$	–	Z		
1			2			3		4		5			6	$\overset{\text{OBLIG}}{\Longrightarrow}$	
1			2			4 + 3		0		5			6		

Condition: $2 = 4$

For example, the sentence 'I saw that the boy that kissed Bella was jealous' is derived as follows:

I saw that	$-[_{NP}$	the boy	$-[_{S}$	that	–	the boy	–	kissed Bella	$]_{S}]_{NP}$ –	was jealous	
1		2		3		4		5		6	$\overset{\text{OBLIG}}{\Longrightarrow}$
1		2		4 + 3		0		5		6	

Condition: 2 = 4

Complex NP Constraint. The first constraint this rule has to obey is the Complex NP Constraint (cf. Subsection 6.1.1). This means that (7.2) should be grammatical, but (7.3) not. In (7.2), the moved phrase 'the vampire' is moved out of the non-complex NP 'a book about ⟨the vampire⟩'; in (7.3), however, 'the vampire' is moved out of the complex NP 'a book which was about ⟨the vampire⟩', which is not allowed.

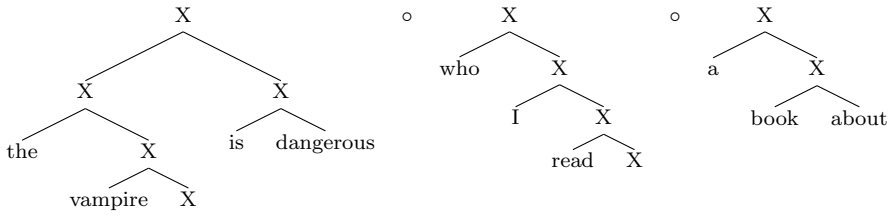
(7.2) the vampire who I read a book about is dangerous

(7.3) * the vampire who I read a book which was about is dangerous

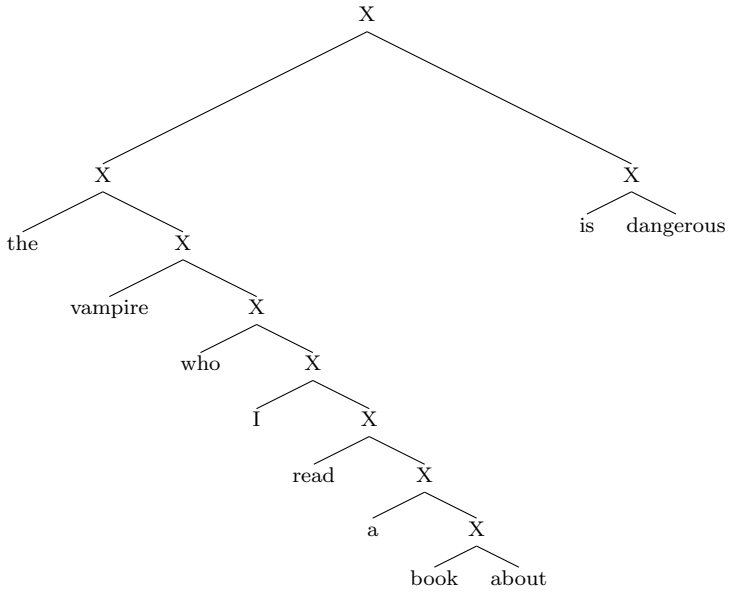
Given the complexity of these sentences, it is not surprising that we ran into memory issues when we tried to let our parser parse them. This is also plausible from a language acquisition modeling point of view: it is unlikely that children are confronted with such sentences, let alone utter them theirselves.

Therefore, we backed off to the second pass of the two-pass model, following the methodology outlined in Subsection 1.2.2. We find the following derivations:

(7.4)

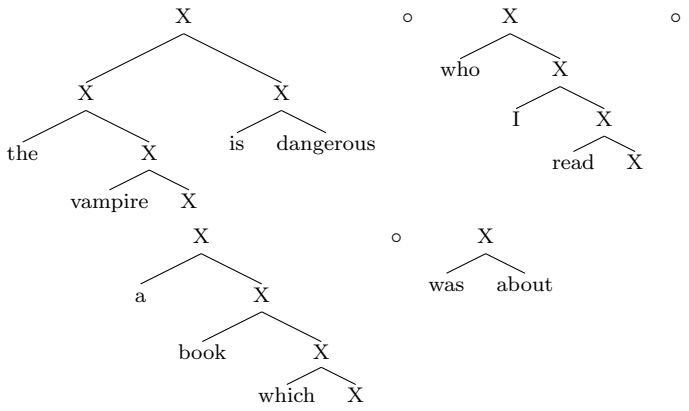


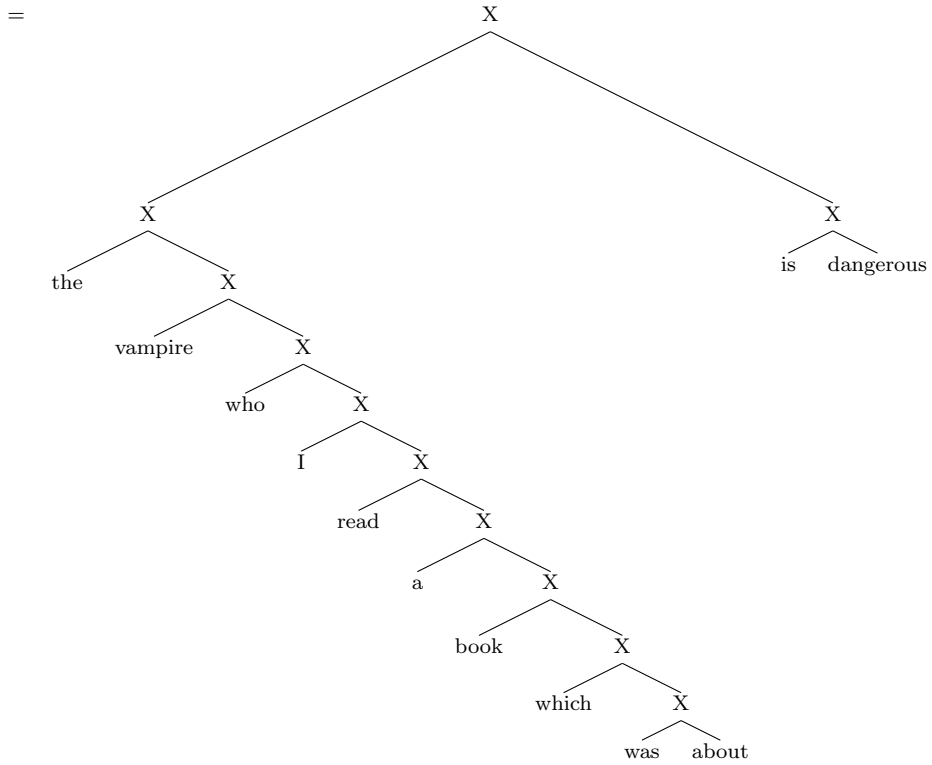
=



(ranking: 1,585,992 + 123,195 + 5,719 = 1,714,906)

(7.5)





(ranking: 1,585,992 + 123,195 + 184,665 + 12,745 = 1,906,597)

We see that the derivation for (7.3) consists of four steps, whereas the derivation for (7.2) consists of only three steps. Hence, the U-DOP approach correctly prefers the latter.[1]

Coordinate Structure Constraint. The second constraint that has to be obeyed is the Coordinate Structure Constraint (cf. Subsection 6.1.2). This means that (7.6) should be grammatical and (7.7) not.

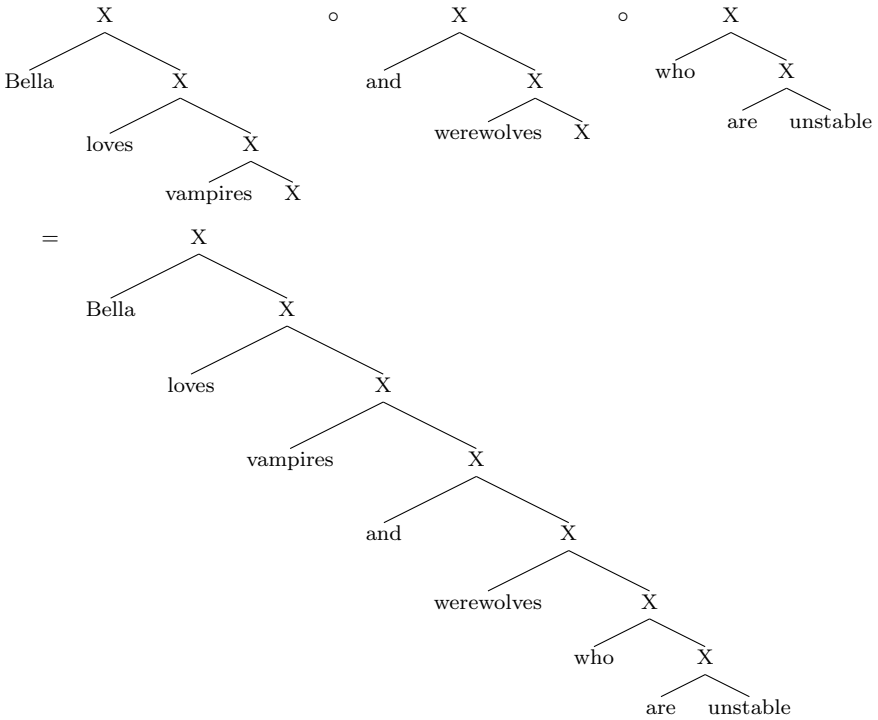
(7.6) Bella loves vampires and werewolves who are unstable

(7.7) * werewolves who Bella loves vampires and are unstable

Again it was not feasible to let our parser parse these sentences; therefore, we backed off to the second pass of the two-pass model and found the following:

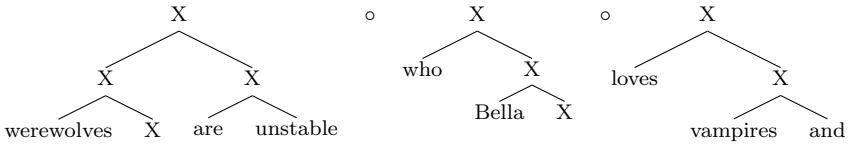
[1] Of course, the ranking score of (7.4) is also better than that of (7.5). However, in this second pass of the two-pass model, we first look at the length of the derivations; the ranking is only used to break ties.

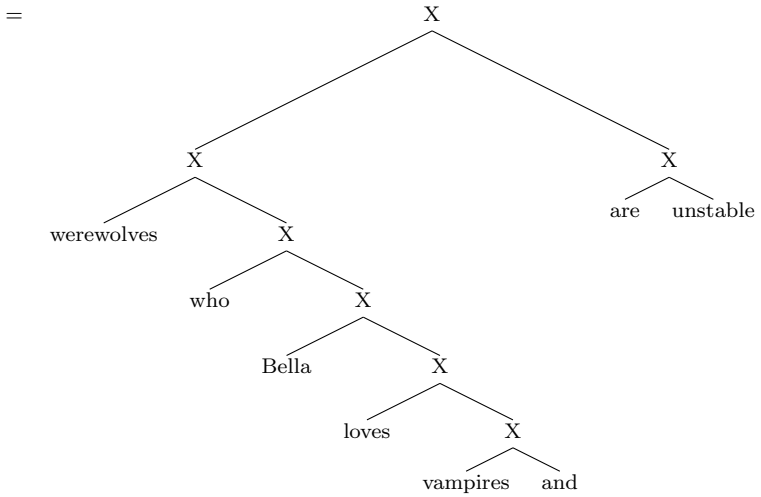
(7.8)



(ranking: 1,264,000 + 209,755 + 132,727 = 1,606,482)

(7.9)





(ranking: 515,341 + 2,325,562 + 169,145 = 3,010,048)

Although both derivations are equally short, we see that the ranking criterion makes the right decision: (7.8) has a lower ranking score and is hence preferred over (7.9).

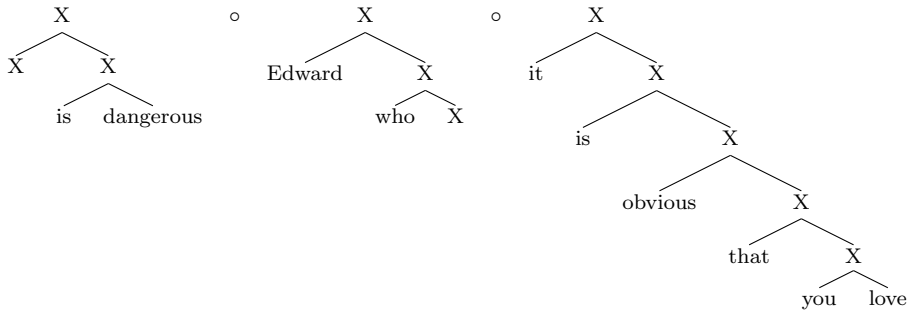
Sentential Subject Constraint. Third, the Sentential Subject Constraint must apply (cf. Subsection 6.1.3). Therefore, (7.10) must be predicted to be more grammatical than (7.11).

(7.10) Edward who it is obvious that you love is dangerous

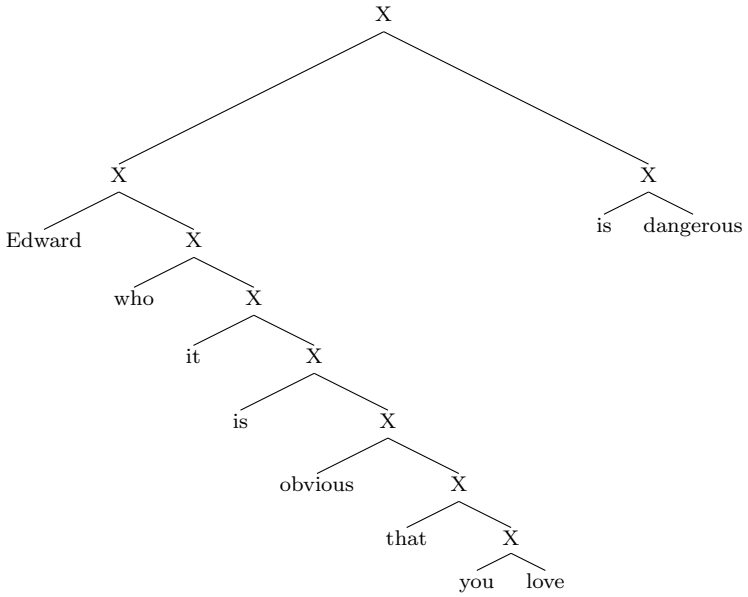
(7.11) * Edward who that you love is obvious is dangerous

Not surprisingly, the parser could not parse these sentences either. Again this is actually cognitively plausible: even adults have to think about (7.10) before they grasp its meaning. We backed off to the second pass of the two-pass model and found the following:

(7.12)

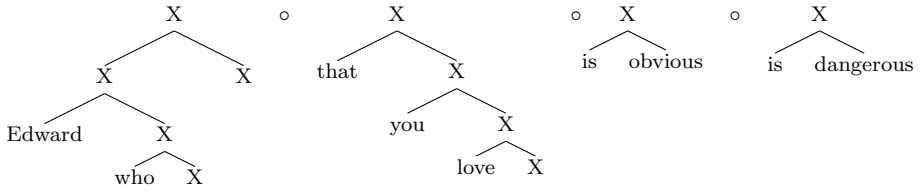


=

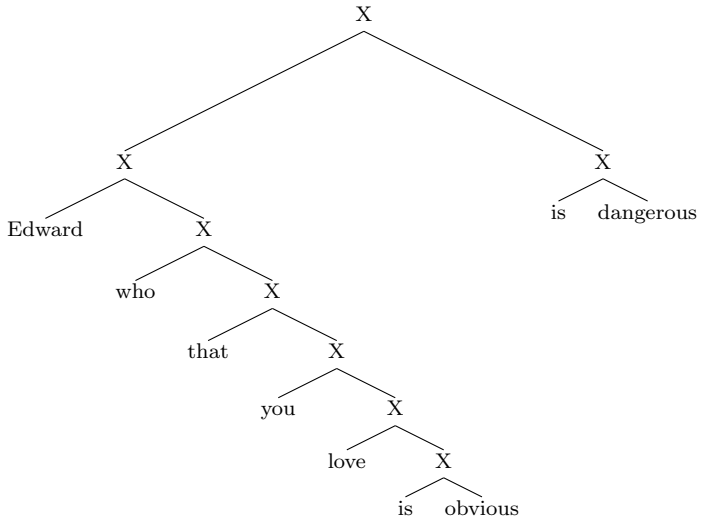


(ranking: 175 + 23,274 + 2,325,562 = 2,349,011)

(7.13)



=



(ranking: 23,274 + 6,444 + 4,085 + 4,085 = 37,888)

We see that the derivation in (7.12) has three steps, whereas the derivation in (7.13) needs four steps. Hence, the U-DOP approach correctly predicts that (7.10) is more grammatical than (7.11).[2]

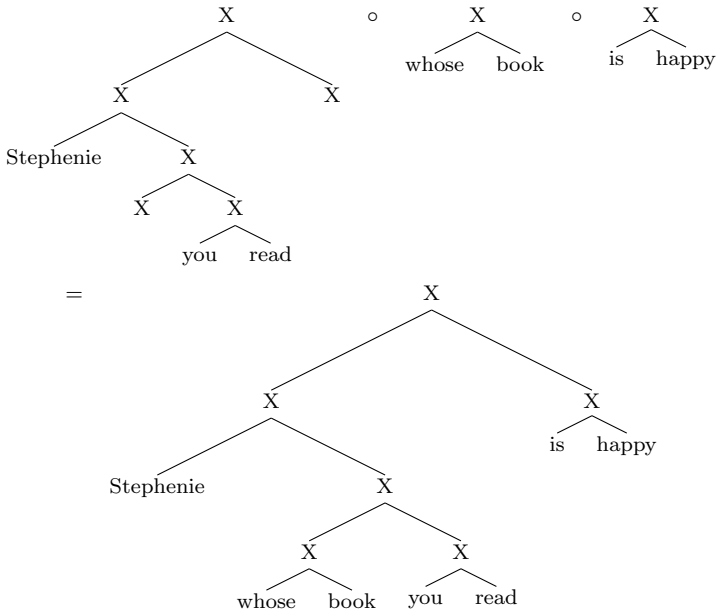
Left Branch Condition. The final constraint the rule has to obey is the Left Branch Condition. According to this condition, (7.14) should be grammatical and (7.15) not.

(7.14) Stephenie whose book you read is happy

(7.15) * Stephenie whose you read book is happy

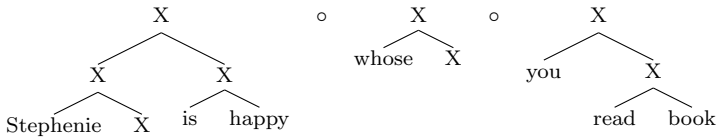
Again, we could not let our parser parse these sentences, so we backed off to the second pass of the two-pass model. We find that both (7.14) and (7.15) have a shortest derivation of three steps (cf. (7.16) and (7.17)). However, there is a clear difference in ranking score: the score of (7.16) is much better than the score of (7.17). Hence, our parser correctly predicts that (7.14) will be preferred over (7.15).

(7.16)

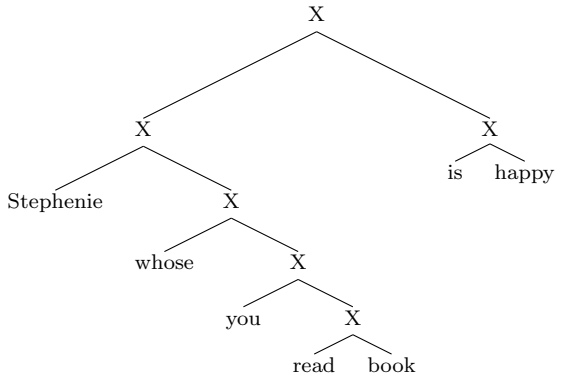


(ranking: 34,424 + 49,493 + 4,085 = 88,002)

(7.17)



[2]Of course, the ranking score of (7.13) is better than that of (7.12); however, in this second pass of the two-pass model, the ranking is only used to break ties.

= 

(ranking: 590,659 + 743 + 14,574 = 605,976)

In this section we have shown that the U-DOP approach that we succesfully applied to wh-questions can also be extended to account for other, similar structures such as relative clauses. In the next sections, we will look at structures which are gradually more dissimilar from wh-questions, and show that U-DOP can still account for them. In this way, we hope to show the generality of the U-DOP approach.

7.2 Extraposition from NP

In Section 6.1, we explained which rules are subject to the island constraints according to Ross: so-called 'chopping' transformations where the chopped constituent is moved over a variable. Now when we look at the rule for Extraposition from NP in (7.18), we see that, formally speaking, this rule must obey the constraints: constituent S (= 2) is chopped and moved over variable Y (= 3).

(7.18) **Extraposition from NP**

X	$[_{NP}$ NP	–	S	$]_{NP}$	–	Y	
	1		2			3	$\overset{\text{OBLIG}}{\Longrightarrow}$
	1		0			3+2	

For example, the sentence 'the book was Stephenie's which I read' is derived from 'the book which I read was Stephenie's as follows:

(7.19) **Extraposition from NP**

$\emptyset$	$[_{NP}$ the book	–	which I read	$]_{NP}$	–	was Stephenie's	
	1		2			3	$\overset{\text{OBLIG}}{\Longrightarrow}$
	1		0			3+2	

So the rule has to obey the constraints. However, according to Ross, this does not explain the difference in grammaticality between (7.20) and (7.21).

(7.20) that Jacob picked Bella up who loves Edward is possible

(7.21) * that Jacob picked Bella up is possible who loves Edward

The Complex NP Constraint cannot explain this, because it applies to *elements* of a sentence dominated by an NP, and here the moved constituent 'who loves Edward' *is* a sentence dominated by an NP. Therefore, a new concept is introduced: 'upward boundedness'.

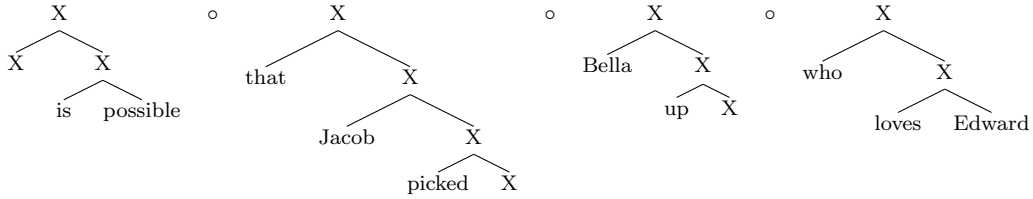
Upward bounded a rule [is] upward bounded if elements moved by that rule cannot be moved over [the boundaries of the first sentence above the elements being operated on] (Ross, 1967, p. 298)

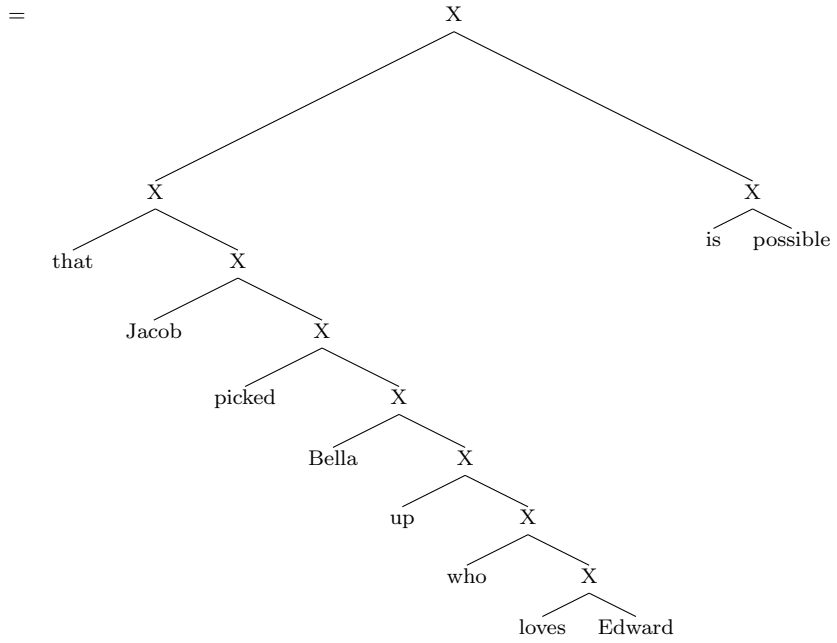
The rule Extraposition from NP is then claimed to be upward bounded, and this explains the difference between (7.20) and (7.21). After all, 'who loves Edward' cannot be moved over the boundaries of "the first sentence above the elements being operated on", i.e. the 'that'-clause. In (7.20), the constituent is merely moved *within* these boundaries, which is allowed; in (7.21), however, it is moved *over* the boundaries, which leads to an ungrammatical sentence.

So to explain the phenomenon of Extraposition from NP, Ross needs additional machinery. We will now show that this is not necessary in the U-DOP account.

As expected, our parser could not parse these sentences due to memory issues. Therefore, we checked the possible derivations manually, and we found the following:

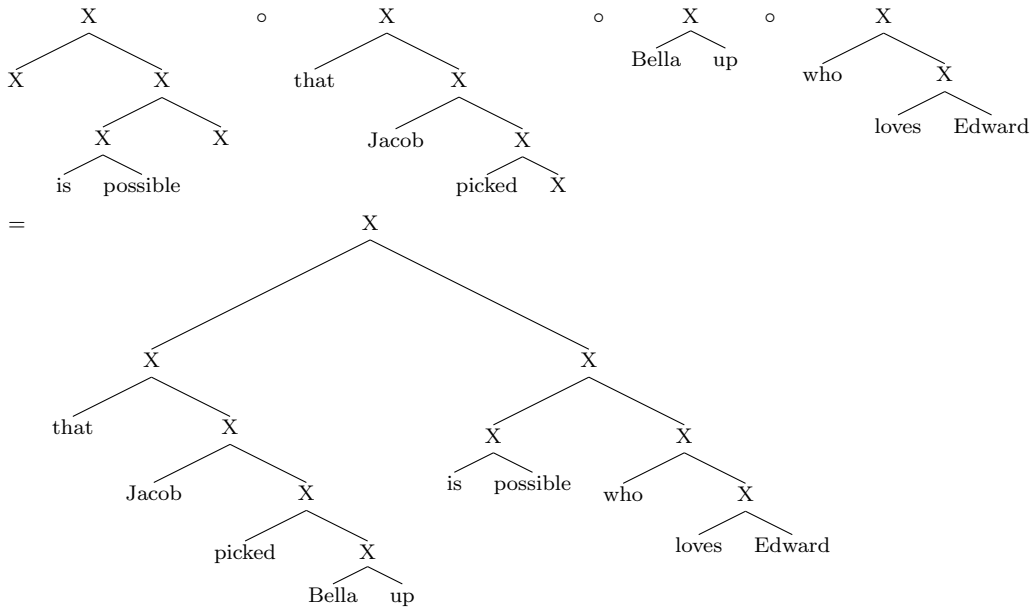
(7.22)





(ranking: 175 + 465,494 + 149,372 + 465,494 = 1,080,535)

(7.23)



(ranking: 3,257 + 465,494 + 176,910 + 465,494 = 1,111,155)

Both derivations consist of four elements, so the ranking needs to break the tie. We see that the ranking of (7.22) is lower than that of (7.23), so the U-DOP approach correctly prefers (7.20) over (7.21).

So we see that U-DOP can account for the problematic instantiations of Extraposition from NP without introducing additional machinery. In this sense, the U-DOP approach is more general than Ross' island constraints.

7.3 Topicalized sentences

In the next two sections, we will discuss the closely related phenomena of topicalization and left dislocation. Interestingly, the former obeys the constraints, but the latter does not. It is this difference that initially probed Ross into distinguishing between 'chopping' and 'copying' transformations: topicalization is a chopping transformation, which needs to obey the constraints; left dislocation is a copying transformation, and hence doesn't need to obey the constraints.

Our goal in these two sections is to show that we do not need to distinguish between these two very similar phenomena: they can be explained with the same mechanisms in the U-DOP framework. Then also in this sense (next to the one in the previous section), our U-DOP approach is more general than Ross' constraints: we can explain more phenomena in a uniform way.

In this section, we will show how U-DOP explains that topicalization seems to obey the island constraints.

Complex NP Constraint. The first constraint that topicalization has to obey is the Complex NP Constraint. Hence, (7.24) should be more grammatical than (7.25).

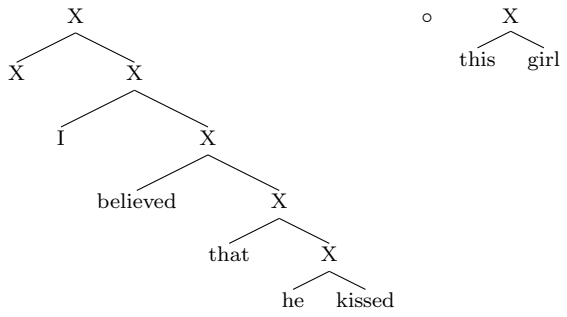
(7.24) this girl I believed that he kissed

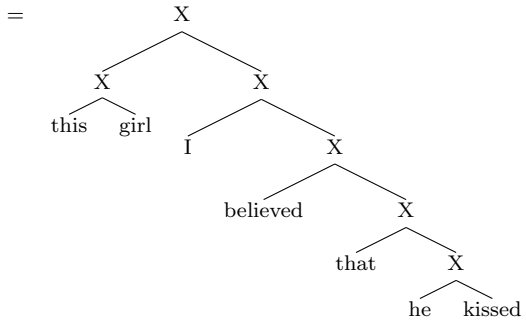
(7.25) * this girl I know the boy who kissed

Of course, our parser was not able to parse these sentences. We think this is also plausible from a language acquisition modeling point of view: it is highly unlikely that children hear this kind of sentences, let alone utter them themselves.

Therefore, we backed off to the second pass of the two-pass model, and found the following:

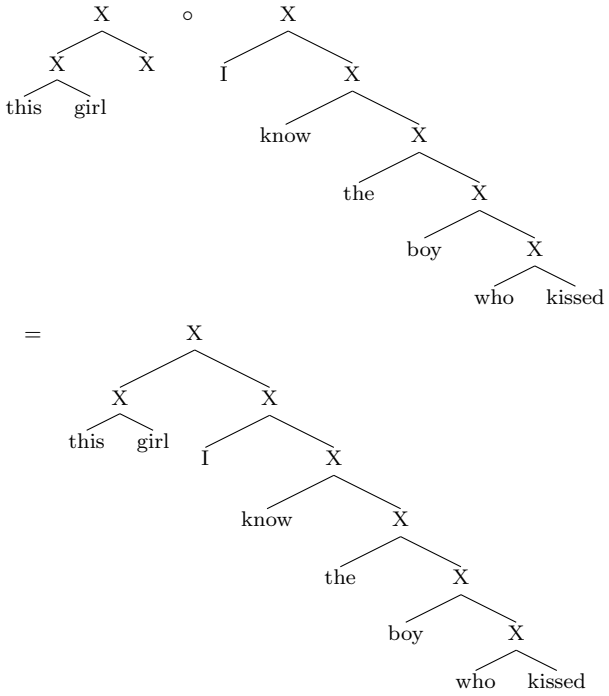
(7.26)





(ranking: 123,195 + 30,164 = 153,359)

(7.27)



(ranking: 8,551 + 2,325,562 = 2,334,113)

We see that both sentences have equally short derivations, but that the ranking of (7.26) is much lower than that of (7.27). Hence, U-DOP correctly prefers (7.24) over (7.25).

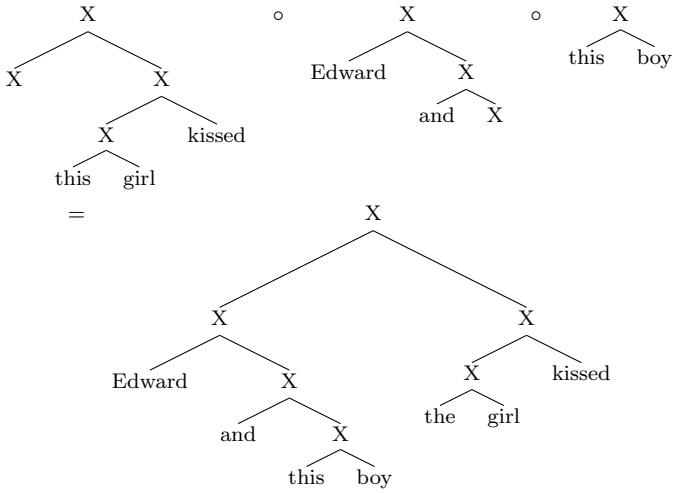
Coordinate Structure Constraint. Second, topicalization must obey the Coordinate Structure Constraint. Hence, (7.28) should be preferred over (7.29).

(7.28) Edward and this boy the girl kissed

(7.29) * this boy the girl kissed Edward and

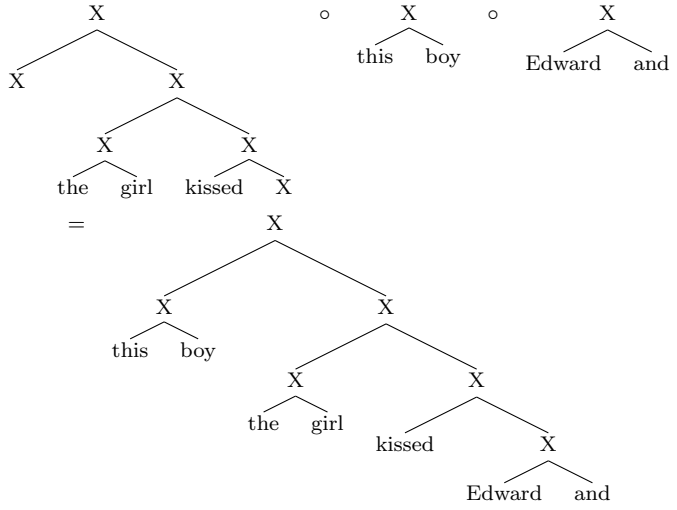
Our parser could not parse these sentences due to memory limitations, and hence, we backed off to the second pass of the two-pass model. We found the following shortest derivations for these sentences:

(7.30)



(ranking: 3,190 + 111,567 + 30,164 = 144,921)

(7.31)



(ranking: 11,687 + 30,164 + 70,422 = 112,273)

Both derivations consist of three elements, and the ranking predicts the wrong alternative to be the more grammatical: (7.31) has a lower (and hence better) ranking than (7.30). In fact, this is basically the same problem as in Subsection 6.3.3: the ranking of the subtree (X noun coordinating conjunction) is better than it should be. We believe that the same solution as in Subsection 6.3.3 would apply here: the new methodology which simulates supervised

parsing would probably assign a much worse (i.e. higher) score to that subtree, causing the entire rank of derivation (7.31) to increase. Note that the difference in rank between both derivations is fairly small, so in that case, the rank of the entire derivation in (7.31) will probably be higher (i.e. worse) than the rank of (7.30), and (7.28) will correctly be preferred over (7.29). Unfortunately, it is not yet possible to investigate this on the basis of the actual Childes-data, since this would involve parsing half of the corpus (around 11,000 sentences), which could not be done in a feasible amount of time.

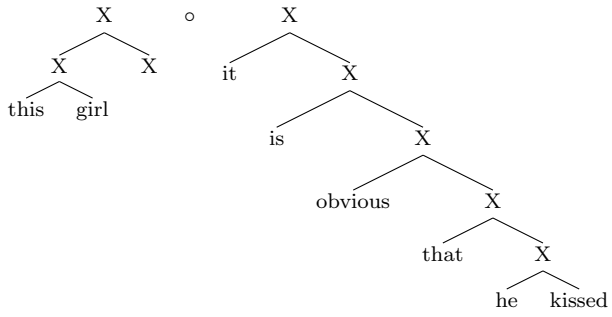
Sentential Subject Constraint. The third constraint the rule of topicalization has to obey is the Sentential Subject Constraint. This means that (7.32) should be more grammatical than (7.33). Note that (7.33) is in fact grammatical, but not with the intended meaning. If the predicate 'obvious' applies to the *girl*, then the sentence is perfectly grammatical. However, if the predicate 'obvious' applies to the *statement* 'that he kissed this girl', then it is ungrammatical.

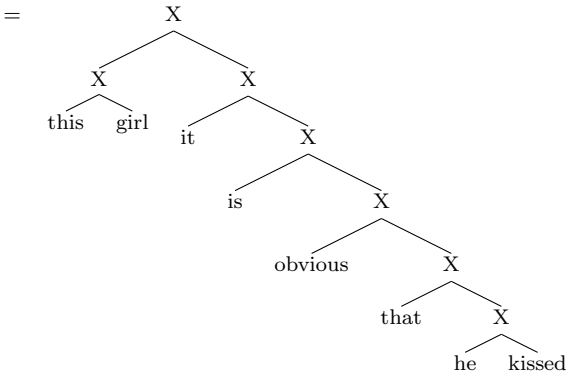
(7.32) this girl it is obvious that he kissed

(7.33) * this girl that he kissed is obvious

Similar to the previous examples of the Sentential Subject Constraint, our parser could not parse these sentences. Therefore, we backed off to the second pass of the two-pass model, and found the derivation in (7.34) for (7.32) and in (7.35) for (7.33).

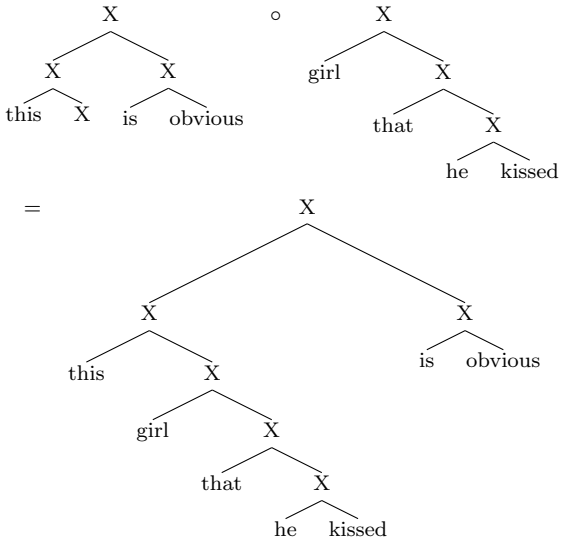
(7.34)





(ranking: 8,851 + 2,325,562 = 2,334,413)

(7.35)



(ranking: 693,021 + 415,940 = 1,108,971)

Both sentences have a shortest derivation of two steps, so we need the ranking to break the tie. We see that (7.35) has a lower, and hence better ranking score than (7.34); so (7.33) will be preferred over (7.32). This is actually not surprising, since it is the *semantics* that make (7.33) ungrammatical, and our parser has no access whatsoever to semantics. Moreover, the parser looks at part-of-speech tags rather than words (we give the words in the examples for ease of exposition), so the derivation for (7.33) is the same as for (7.36), which is in fact fairly grammatical. Note that using words instead of part-of-speech tags alone would not solve the problem; a semantic component is really needed to explain the difference in acceptability. For example, (7.36) is grammatical when 'nice' is applied to the girl, but not when 'nice' is applied to the statement, meaning 'it is nice of him to kiss this girl'.

(7.36) this girl that he kissed is nice

So the U-DOP approach cannot explain the difference in grammaticality between (7.32) and (7.33). However, this problem has more to do with semantics than syntax, so we do not consider it a disadvantage of the U-DOP account.

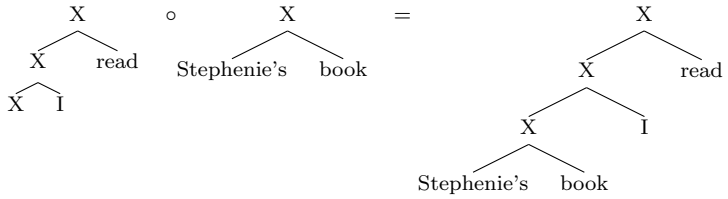
Left Branch Condition. Finally, topicalization has to obey the Left Branch Condition. According to this condition, (7.37) should be grammatical and (7.38) not.[3]

(7.37) Stephenie's book I read

(7.38) * Stephenie's I read book

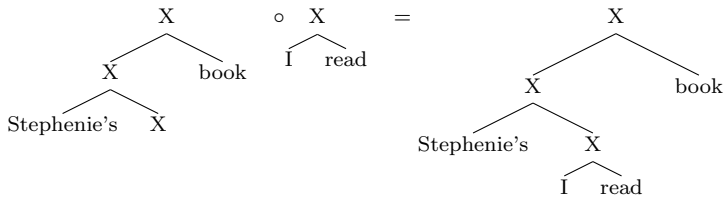
These sentences our parser could parse. We found the following derivations:

(7.39)



(ranking: 608 + 2,784 = 3,392)

(7.40)



(ranking: 3,139 + 488 = 3,627)

Both sentences have a shortest derivation of two elements, but the ranking score of (7.39) is better than that of (7.40). Hence, the parser correctly prefers (7.37) over (7.38).

In this section, we have shown that the U-DOP account can be used to explain the behavior of sentences with topicalization. In the next section, we will investigate whether this account is more general than the Ross' constraints, and can account for the highly similar phenomenon of left dislocation.

[3]In the Childes corpus, possessives such as 'Stephenie's' are analyzed as one word, with part-of-speech tag 'n(:prop)|cat-POSS'.

7.4 Left dislocation

In this section, we will investigate whether the U-DOP approach is general enough to also capture phenomena that lie outside the scope of Ross' islands constraints, such as left dislocation.

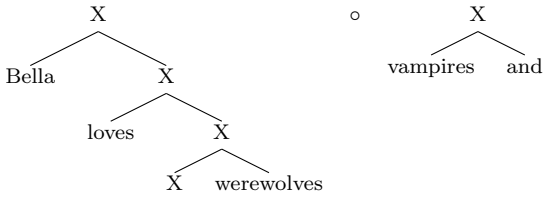
In the first place, an account of left dislocation must show that the island constraints do not apply. For frameworks with an absolute notion of grammaticality, this is difficult: they have to show why some sentences which are thought to be ungrammatical, turn out to be grammatical after all. However, for approaches such as U-DOP, which honor a *relative* notion of grammaticality, this is trivial: all sentences are grammatical to some degree (because all sentences can be generated), so also the sentences violating the presumed constraints (because also these sentences can be generated).[4]

When we let our parser parse (7.41) and (7.42), for example, we see that the sentence violating the Coordinate Structure Constraint (7.42) can be derived with as few steps as the 'regular' sentence, and even has a better ranking score![5]

(7.41) Bella loves vampires and werewolves

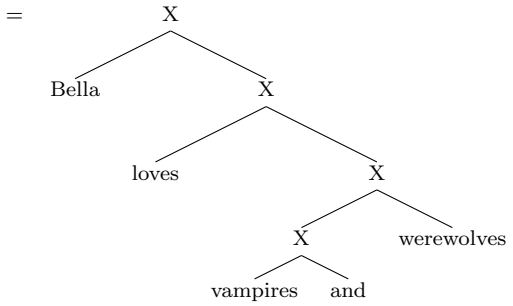
(7.42) werewolves, Bella loves vampires and them

(7.43)



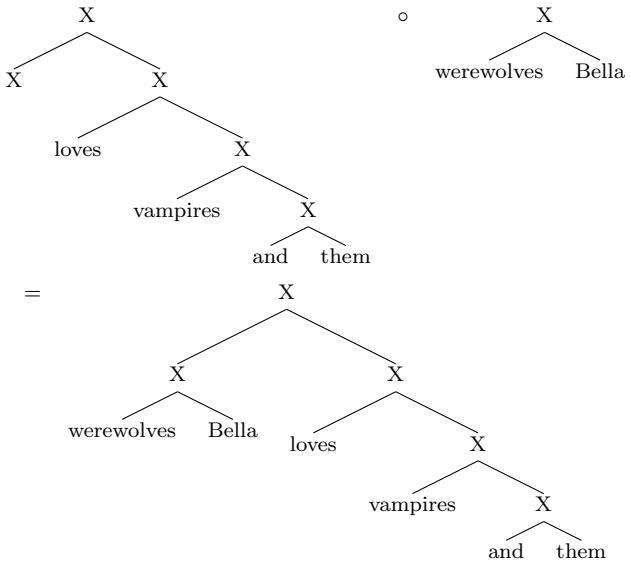
[4] We are aware that this may seem too simplistic as an explanation. However, as we noted in the discussion of U-DOP's *generating* capacity at the end of Section 2.3, this idea that all sentences can be generated shifts the weight of the workload to the sentences that are generally considered to be unacceptable. The greatest challenge for U-DOP lies in explaining why humans judge certain sentences to be less acceptable than others. So where we can simplistically imagine the workload for a nativist linguist to be 50% explaining grammatical sentences and 50% explaining ungrammatical sentences, the workload for the U-DOP approach is 100% explaining sentences that are considered to be ungrammatical: it gets the grammatical sentences for free, but has to work harder for the others.

[5] Of course, it might be considered a disadvantage of the U-DOP approach that it now predicts that (7.42) is slightly more grammatical than (7.41).



(ranking: 70,422 + 9,392 = 79,814)

(7.44)



(ranking: 46,902 + 18,150 = 65,052)

Since accounting for the fact that left dislocation does not obey the Coordinate Structure Constraint means showing that sentences violating it, such as (7.42), are grammatical, the U-DOP account can easily do this. In the first place, the U-DOP account has no difficulty whatsoever with explaining why sentences are grammatical, i.e. can be generated (since all sentences can be generated). In the second place, because such sentences also seem to be easily derivable, with a good ranking score, and hence high up the grammaticality scale.

Showing that sentences are grammatical is rather trivial for U-DOP (although it is a hard task for the nativist approaches). More difficult is explaining why some sentences are considered ungrammatical. For a complete account of left dislocation, we should therefore also explain the restrictions on this construction.

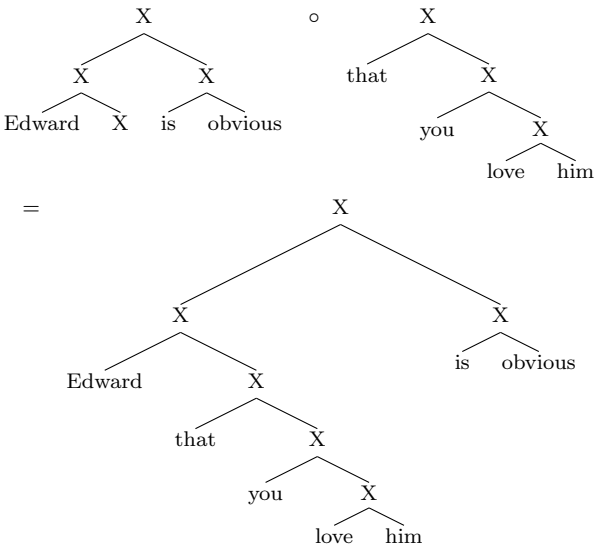
A restriction that is mentioned (but not explained) by Ross, is the fact that the moved constituent must be moved to the left of the *main* clause; movement merely to the left of a subordinate clause results in an ungrammatical sentence. For example, (7.45) is grammatical, because 'Edward' is moved to the left of the main clause;[6] (7.46), on the other hand, is ungrammatical, because 'Edward' is only moved to the left of the subordinate clause 'that you love ⟨Edward⟩'.

(7.45) Edward, that you love him is obvious

(7.46) * that Edward, you love him is obvious

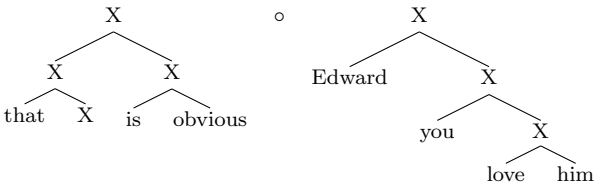
Our parser could not parse these sentences due to memory issues, so we backed off to the second pass of the two-pass model. We found the derivation in (7.47) for (7.45) and in (7.48) for (7.46).

(7.47)

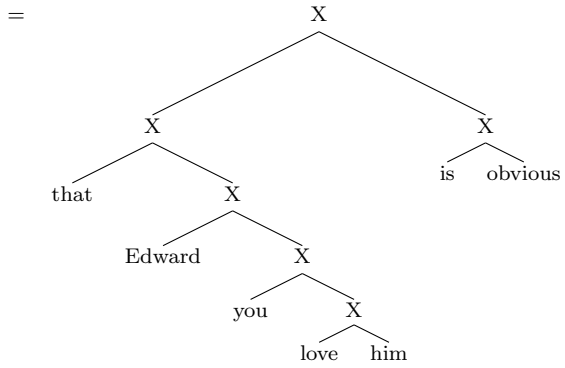


(ranking: 590,659 + 57,785 = 648,444)

(7.48)



[6]Note that the grammaticality of (7.45) also shows that the Sentential Subject Constraint does not apply to left dislocation.



(ranking: 876,625 + 415,940 = 1,292,565)

We see that both derivations are equally short, but the ranking chooses the correct alternative: (7.47) has a lower ranking than (7.48), so (7.45) is preferred over (7.46).

So we see that U-DOP can both account for the fact that left dislocation does not obey the island constraints, as for the constraints that left dislocation *does* have to obey. Therefore, the U-DOP approach is more general than the nativist approach: rather than postulating constraints, and then making a distinction between constructions obeying them and constructions that do not obey them, U-DOP can account for all kinds of constructions in a uniform way.

7.5 Conclusion

In this chapter, we have seen how the U-DOP approach to language acquisition is more general than the nativist approach, as expressed in Ross' island constraints. First, we have shown how the U-DOP approach is at least equally general, by showing how it can, similarly to the islands constraints, easily be extended to other constructions such as relative clause formation.

Then we have seen an example where Ross' island constraints do not suffice to capture all facets of a phenomenon: to explain the restrictions on Extraposition from NP Ross needed the additional notion of 'upward boundedness'. We have shown that the U-DOP approach has no need for such an extension and can explain the problematic cases for Extraposition from NP in the same way as it explained all the other phenomena.

Finally, we looked at two closely related phenomena: topicalization and left dislocation. The former obeys the constraints, the latter does not. To explain this difference in behavior, Ross makes a distinction between chopping and copying transformations: the former obey the constraints, the latter do not. We showed that the U-DOP approach can account for both phenomena without distinguishing between them, and can also capture other aspects of left dislocation, other than that it does not obey the constraints. So the U-DOP approach is shown to be more general in several ways: it can account for the fact that some transformations obey the constraints, for the fact that some

transformations do not obey the constraints, and for additional facts about the latter transformations.

Remarkable is that in this chapter the parser was often incapable of parsing the sentences under investigation, and that we had to resort to the second pass of the two-pass model quite a few times. We believe this is not a disadvantage of the U-DOP approach per se. In the first place, this difficulty is merely due to practical reasons —it is not the case that the parser is *in principle* unable to parse these problematic sentences. So in the future, when the hardware will be more advanced, and computers will have more working memory, the parser *will* be able to parse these sentences. After all, the second pass of the two-pass model already shows that it is in principle possible to find derivations for these sentences.

Second, we believe that this difficulty with parsing is also cognitively plausible. If we want the parser to model the way a child would process these sentences, it is only to be expected that it will have difficulty with the examples in this chapter: also a child would have difficulty processing these far-fetched sentences that are rarely encountered in speech. So in a certain sense, the effort the parser has with these sentences is in line with the effort a child would have. Therefore, the parser models language acquisition more accurate than the nativist approach: in the nativist approach differences in processing difficulty are not reflected at all.

In sum, in this chapter we have shown that (i) the U-DOP approach is more general than the nativist approach with Ross' islands constraints, and (ii) that the sometimes far-fetched examples that the nativist framework comes up with are equally difficult for our parser as they would be for a child, so that the parser more accurately models language acquisition with respect to processing difficulties.

Chapter 8

Conclusion

This concluding chapter consists of two sections. In the first section, we discuss the results obtained in this thesis, both of a computational and a theoretical nature. In the second section, we look at some questions for further research that arose throughout the thesis.

8.1 Results

Recall that the goals of this thesis were twofold: on the one hand, to contribute to the field of unsupervised parsing, by developing implementations of U-DOP; on the other hand, to refute Arguments from Poverty of Stimulus by using those implementations to show that certain language phenomena *can* be accounted for in an empirical way.

The first goal has certainly been achieved. Already the basic implementation of U-DOP (Section 3.1) obtains results on the ATIS-corpus that are better than the state-of-the-art. We contributed two innovations to the field of unsupervised parsing. The first innovation, doing syntactic category labeling and parsing simultaneously (Section 3.3), did not yet show promising results (only a slight improvement on the basic implementation), but we think this is due to the small corpus on which the evaluation was carried out. The second innovation, a new methodology for unsupervised parsing (Section 3.4), turned out to be a great success: although its training phase took somewhat longer, its parsing phase was extremely fast (0.02 seconds per sentence). This is the behavior that we want from an NLP-application: training (which only needs to be done once) may be slow, as long as the parsing (which needs to be done often) goes fast. Moreover, this new methodology achieved by far the best results on the ATIS-corpus (61.2%, compared to Klein (2005)'s 51.2%). Furthermore, these results were achieved with an inferior version of the algorithms; this gives a promising outlook for when evaluation can be carried out with the full versions.

With respect to the second goal, we have carried out an in-depth study of the phenomenon of wh-questions. First, we showed the involved machinery that

the nativist account (as formulated in Ross (1967) and Adger (2003)) needs to explain all details of this construction. Next, we showed how the U-DOP approach can capture all these details in a simple and uniform way in most of the cases. Hence, we have shown that at least for this phenomenon, Arguments from Poverty of Stimulus can no longer be invoked: step (ii), where it is claimed that children cannot learn the phenomenon on the basis of input alone, is now refuted. Furthermore, we have shown that the U-DOP approach can easily be extended to cover other phenomena, even phenomena that fall out of the scope of the traditional nativist account developed in Ross (1967). In Table 8.1, an overview is given of all the phenomena discussed in this thesis, and whether they can be succesfully explained in the U-DOP framework.

So we demonstrated that the U-DOP approach, which is conceptually much simpler, is as adequate as the involved nativist accounts, and even more general: more phenomena can be captured in a uniform way.

8.2 Questions for further research

Both in the computational (Part I) and the theoretical component (Part II) of this thesis several questions for further research arose.

In the computational component, the immediate first concern is to build more efficient implementations. This is necessary to be able to perform further evaluation, especially for the Labeling+Parsing-algorithm, but also to be able to use them more for theoretical questions: quite often, we had to back off to the second pass of the two-pass model in the theoretical part of this thesis, because the parser could not analyze the sentences due to memory issues.

Second, the delicate interplay between frequency (ranking) and maximal structural overlap (shortest derivation) merits further exploration. It is clear that both should play a role, but it should be further investigated how the interaction should work. Guided by the F1-scores obtained by the different implementations, we hypothesize that maximal structural overlap will play the largest role: the implementations on the basis of the shortest derivation (basic) mostly scored better than the implementations based on the ranking (optim).

In the theoretical component, the first concern is also to have more efficient implementations, so that more phenomena can be investigated in the first pass of the two-pass model (i.e. by using an implementation), rather than having to back off to the second pass.

Second, it is desirable to add a *semantic* component to the framework. As we saw in Section 7.3, the unacceptability of some sentences can only be explained by taking semantics into account. This has already been done in Bod and Kaplan (2003), but further work definitely needs to be conducted in this area.

Finally, the work done in this thesis is only a first step to investigate phenomena that are used in Arguments from Poverty of Stimulus. Further work must certainly look at other phenomena (e.g. binding, NPI's, ...) that are traditionally used in these arguments, and investigate whether it is indeed the case that they cannot be learned from input alone.

Table 8.1: An overview of all phenomena discussed in this thesis. The column 'Method' indicates whether the first or the second pass of the two-pass model was used, and the column 'Succesful?' indicates if the U-DOP approach was succesful, or how it could be amended.

Phenomenon	Method	Succesful?
Subject Auxiliary Inversion (5.2)	2nd pass	yes
WH-Questions		
Unbounded Scope (6.3.1)	2nd pass	yes
Complex NP Constraint (6.3.2)	2nd pass	interplay shortest derivation/ranking
Coordinate Structure Constraint (6.3.3)	1st pass	new approach (3.4)
Left Branch Condition (6.3.5)	1st pass	yes
Left Branch Condition 'how'	1st pass	more fine-grained POS-tags
Subject WH-questions (6.3.6)	1st pass	yes
WH in situ (6.3.7)	1st pass	yes
Superiority (6.3.8)	1st pass	yes
Extended Superiority	2nd pass	yes
No Superiority	1st pass	yes
Embedded WH-questions (6.3.9)	2nd pass	yes
WH-islands (6.3.10)	1st pass	yes
Relative Clause Formation (7.1)		
Complex NP Constraint	2nd pass	yes
Coordinate Structure Constraint	2nd pass	yes
Sentential Subject Constraint	2nd pass	yes
Left Branch Condition	2nd pass	yes
Extraposition from NP (7.2)	2nd pass	yes
Topicalization (7.3)		
Complex NP Constraint	2nd pass	yes
Coordinate Structure Constraint	2nd pass	new approach (3.4)
Sentential Subject Constraint	2nd pass	semantics
Left Branch Condition	1st pass	yes
Left Dislocation (7.4)		
Coordinate Structure Constraint	1st pass	yes
Restriction	2nd pass	yes

Bibliography

D. Adger. *Core syntax: A minimalist approach.* Oxford University Press, 2003.

M. Bansal and D. Klein. Simple, accurate parsing with an all-fragments grammar. In *Proceedings of the 48th annual meeting of the Association for Computational Linguistics*, pages 1098 – 1107. Association for Computational Linguistics, 2010.

C. Biemann. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the COLING/ACL 2006 Student Research Workshop*, pages 7 – 12. Association for Computational Linguistics, 2006.

R. Bod. A computational model of language performance: Data Oriented Parsing. In *Proceedings of the 14th conference on Computational Linguistics*, volume 3, pages 855 – 859. Association for Computational Linguistics, 1992.

R. Bod. Using an annotated language corpus as a virtual stochastic grammar. In *Proceedings of the national conference on Artificial Intelligence*, page 778, 1993.

R. Bod. Parsing with the shortest derivation. In *Proceedings of the 18th conference on Computational Linguistics*, volume 1, pages 75 – 81. Association for Computational Linguistics, 2000.

R. Bod. Exemplar-based syntax: How to get productivity from examples. *The Linguistic Review*, 23:291 – 320, 2006.

R. Bod. Is the end of supervised parsing in sight? In *Proceedings of the 45th annual meeting of the Association for Computational Linguistics*, pages 400 – 407. Association for Computational Linguistics, 2007.

R. Bod. From exemplar to grammar: A probabilistic analogy-based model of language learning. *Cognitive Science*, 33:752 – 793, 2009.

R. Bod and R. Kaplan. A DOP model for Lexical-Functional Grammar. In R. Bod, K. Sima'an, and R. Scha, editors, *Data-Oriented Parsing*, pages 211 – 232. CSLI, 2003.

R. Bod, K. Sima'an, and R. Scha, editors. *Data-Oriented Parsing.* CSLI, 2003.

P. Brakel and M. Smets. Using the shortest derivation for parsing with U-DOP. Unpublished manuscript, 2009.

R. Brown. *A first language: the early stages*. Harvard University Press, 1973.

J. Bybee. From usage to grammar: The mind's response to repetition. *Language*, 82:711 – 733, 2006.

N. Chomsky. *The minimalist program*. MIT Press, 1995.

N. Chomsky. *The architecture of language*. Oxford University Press, 2000.

N. Chomsky, M. Hauser, and W. Fitch. The faculty of language: What is it, who has it, and how did it evolve? *Science*, 298:1569 – 1579, 2002.

A. Clark. Unsupervised induction of stochastic context-free grammars using distributional clustering. In *Proceedings of the 2001 workshop on Computational Natural Language Learning*, pages 1 – 8, 2001.

S. Crain and R. Thornton. Acquisition of syntax and semantics. In M. Traxler and M. Gernsbacher, editors, *Handbook of Psycholinguistics*. Elsevier, 2006.

S. Evert and A. Lenci. Distributional semantic models, 2009. Course taught at ESSLLI '09, Bordeaux, France.

D. Gentner. Structure mapping in analogy and similarity. *American Psychologist*, 52:45 – 56, 1997.

A. Goldberg. Constructions: A new theoretical approach to language. *Trends in Cognitive Sciences*, 7:219 – 224, 2003.

J. Goodman. Parsing algorithms and metrics. In *Proceedings of the 34th annual meeting of the Association for Computational Linguistics*, pages 177 – 183. Association for Computational Linguistics, 1996.

J. Goodman. Efficient parsing of DOP with PCFG-reductions. In R. Bod, K. Sima'an, and R. Scha, editors, *Data-Oriented Parsing*, pages 125 – 146. CSLI, 2003.

L. Huang and D. Chiang. Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 53 – 64. Association for Computational Linguistics, 2005.

D. Klein. *The unsupervised learning of natural language structure*. PhD thesis, Stanford University, 2005.

B. MacWhinney. A multiple process solution to the logical problem of language acquisition. *Journal of Child Language*, 3:883 – 914, 2004.

C. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999.

M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19:313 – 330, 1993.

G. Pullum and B. Scholz. Empirical assessment of stimulus poverty arguments. *The Linguistic Review*, 19:9 – 50, 2002.

R. Reichart and A. Rappoport. Unsupervised induction of labeled parse trees by clustering with syntactic features. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 721 – 728, 2008.

J. Ross. *Constraints on variables in syntax.* PhD thesis, Massachusetts Institute of Technology, 1967.

R. Scha. Taaltheorie en taaltechnologie; competence en performance. In Q. de Kort and G. Leerdam, editors, *Computertoepassingen in de Neerlandistiek*, pages 7 – 22. Landelijke Vereniging van Neerlandici, 1990.

H. Schütze. Distributional part-of-speech tagging. In *Proceedings of the 7th conference of the European chapter of the Association for Computational Linguistics*, pages 141 – 148, 1995.

M. Smets. A DOP-inspired approach to syntactic category induction. Unpublished manuscript, 2010.

M. Tomasello. *Constructing a language: A usage-based theory of language acquisition.* Harvard University Press, 2003.

M. van Zaanen and P. Adriaans. Alignment-based learning versus Emile: a comparison. In *Proceedings of the Belgian-Dutch Conference on Artificial Intelligence*, pages 315 – 322, 2001.

H. Waterfall, B. Sandbank, L. Onnis, and S. Edelman. An empirical generative framework for computational modeling of language acquisition. *Journal of Child Language*, 37:671 – 703, 2010.

J. Zuidema. *The major transitions in the evolution of language.* PhD thesis, University of Edinburgh, 2005.

Appendix A

POS-tags of the ATIS corpus

(We disregarded all punctuation information.)

CC	coordinating conjunction
CD	cardinal number
DT	determiner
EX	existential *there*
FW	foreign word
IN	preposition or subordinating conjunction
JJ	adjective
JJR	adjective, comparative
JJS	adjective, superlative
LS	list item marker
MD	modal
NN	noun, singular or mass
NNS	noun, plural
NNP	proper noun, singular
NNPS	proper noun, plural
PDT	predeterminer
POS	possessive ending
PRP	personal pronoun
PRP$	possessive pronoun
RB	adverb
RBR	adverb, comparative
RBS	adverb, superlative
RP	particle
SYM	symbol
TO	*to*

UH	interjection
VB	verb, base form
VBD	verb, past tense
VBG	verb, gerund or present participle
VBN	verb, past participle
VBP	verb, non-3rd person singular present
VBZ	verb, 3rd person singular present
WDT	wh-determiner
WP	wh-pronoun
WP$	possessive wh-pronoun
WRB	wh-adverb

Appendix B

POS-tags of the Childes corpus

(We disregarded all punctuation information.)

adj	adjective
adv	adverb
adv:int	adverb, intensifying
adv:wh	adverb, question
aux	auxiliary
co	communicator
conj:coord	conjunction, coordinating
conj:subord	conjunction, subordinating
conj:prag	conjunction, pragmatic
det	determiner
det:poss	determiner, possessive
inf	infinitive marker (*to*)
n	noun
n:prop	noun, proper
n:adv	noun, adverbial
num	number, cardinal
ptl	particle
prep	preposition
pro	pronoun, personal
pro:refl	pronoun, reflexive
pro:poss	pronoun, possessive
pro:dem	pronoun, demonstrative
pro:indef	pronoun, indefinite
pro:exist	pronoun, existential
pro:wh	pronoun, question

qn	quantifier
v	verb
part	verb, present participle
v:cop	verb, copula