

University of Warsaw
Faculty of Philosophy

Magdalena Borysiak
Record book number: 446267

Dependency structure of English
coordination: a surface-syntactic
approach

Bachelor's thesis
in the field of Cognitive Science

The thesis was written under the supervision of
prof. dr hab. Adam Przepiórkowski
Faculty of Philosophy, University of Warsaw
Institute of Computer Science, Polish Academy of Sciences

Warsaw 2024

Summary

Key words

Streszczenie

Słowa kluczowe

Contents

1	Introduction	4
2	Theoretical background	6
2.1	Dependency grammars	6
2.2	Dependency length minimization	7
2.3	Possible dependency structures of a coordination	8
2.3.1	Bouquet/Stanford	8
2.3.2	Chain/Moscow	9
2.3.3	Multi-headed/London	10
2.3.4	Conjunction-headed/Prague	11
2.4	Previous studies	12
2.5	Universal Dependencies	13
2.6	Surface-syntactic Universal Dependencies	14
2.6.1	Criteria for choosing heads of dependencies	14
2.6.2	Explicit information about shared dependencies	16
2.6.3	Learnability of dependency schemes	17
3	Data processing	19
3.1	Parser training	19
3.2	Data extraction	20

Chapter 1

Introduction

The aim of this work is to expand on already existing research on the coordinate structures in the English language. An example of a coordination from the Corpus of Contemporary American English (Davies, 2023) (COCA, henceforth) is given in (1) – the word *and* serves as the conjunction, *this* and *so many things* as the conjuncts.

- (1) We’re opposites *in* [[this] and [so many things]].

This coordination is consistent with the observation that in English often the conjuncts on the left of a coordination are shorter than the ones on the right. According to Przepiórkowski and Woźniak (2023) however, the placement of the coordination’s governor (the head of the whole structure, the word *in* in the example above) might have an influence on the way the coordination is structured: if the governor is on the left (as in (1)) or absent from the coordination altogether (as in (2)), then the left conjunct will be shorter. That is not necessarily the case if the governor is on the right (as in (3)).

- (2) [[She was right], but [he didn’t want to say so]].

- (3) [[Cuban flags unfurled from windows] and [women wept]], Perez says.

The Dependency Length Minimization effect is proposed as an explanation for this, which paired with the results of the research provides an argument for symmetric theories of coordination and against asymmetric ones. There were however some limitations to the research that needed to be addressed. The study was based on the Penn Treebank, a corpus already annotated with constituency structures. The presence of a manually approved syntactic annotation is a huge advantage for research like this, but unfortunately the corpus describes only a narrow fragment of the English language – it is made up exclusively of the material from the Wall Street Journal and consists of 1.25M tokens, among which there were only 21,825 coordinations to analyse.

A replication of those results using a different resource has already been attempted and described in an unpublished manuscript by Przepiórkowski et al. (2023). The results were similar, but more precise – they supported not symmetric approaches to coordination in general, but specifically the multi-headed one. The corpus used there was COCA, which consists of over one billion words found in eight different genres ranging from academic to spoken. Unfortunately, it does not have any syntactic annotation that would immediately

allow for data extraction and analysis similar to those in the study described above – it had to be annotated automatically. Because of that, the quality of the data was significantly lower – only 50.1% of the coordinations included in the evaluation of data were parsed and extracted correctly.

Issues with the data could possibly appear either at the parsing stage or at the extraction stage, and using Surface Syntactic Universal Dependencies (SUD) could help with both of those. According to Tuora et al. (2021), for some parsers (particularly the graph-based ones) the SUD annotation scheme might be easier to learn than Universal Dependencies – the annotation scheme used in the replication study mentioned above. As for the issues at the extraction stage, the SUD scheme has some structural benefits, that can be used to create more accurate heuristics.

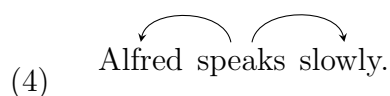
Work reported in this thesis is based on COCA annotated automatically using Stanza (Qi et al., 2020), a graph-based dependency parser, in accordance with the SUD annotation scheme. The structure of this thesis is the following: Chapter 2 describes in more detail some of the theoretical aspects mentioned in the current chapter – this includes previous findings on coordinations in English, the Dependency Length Minimization effect and how it corresponds to different annotation schemes used in dependency treebanks, relevant differences between the UD and SUD annotation schemes and why the syntactic scheme seems to be more appropriate here. Chapter 3 describes how the corpus data was prepared for analysis – from training the parser to extracting the information about coordinate structures. Chapter 4 presents the analysis of the data, which is discussed in Chapter 5. Finally Chapter 6 covers the limitations of this research.

Chapter 2

Theoretical background

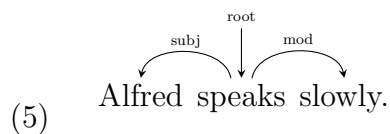
2.1 Dependency grammars

The first full-fledged dependency grammar was proposed by Tesnière (2015). One of the key ideas included in his work, was that a sentence is not comprised solely of its words, but also of the connections between them – dependencies. The connections he proposed were directed, therefore one of the words is always a governor (head) and the other is a dependent. Another crucial element of Tesnière’s approach was verb centrality – the verb is the root of every sentence structure in dependency grammar.



Tesnière as an example used the sentence *Alfred speaks slowly*, for which a dependency tree is shown in (4). It visualises the rules described above – all of the words in a sentence are connected, those connections are directed and the verb is central the whole structure.

Dependency grammars have changed significantly since Tesnière’s ideas were published. One example of such changes is the widespread usage of dependency labels, which describe the grammatical function that a word serves – Tesnière differentiated only between actants and circumstants, which can be understood as obligatory dependencies of a verb, that complete its meaning, and optional dependencies, that are not necessary to complete the meaning of the verb. Different corpora have their own ideas for sets of dependency labels, for instance full annotation for (4) could look similarly to (5), with the label **root** marking the central element of the sentence, **subj** marking the subject of the sentence and **mod** marking a modifier of the verb.





Section 2.2 describes some specific phenomena that can be explained using dependency grammars. In Section 2.3 there are described some of the ideas for dependency annotation of coordinate structures that have been used in

different corpora and in Section 2.4 the studies, that this one is based on, are described. The last two sections of this chapter delve into more detail about two projects concerned with creating consistent dependency annotation schemes – Universal Dependencies in Section 2.5 and Surface-syntactic Universal Dependencies in 2.6.

2.2 Dependency length minimization

Familiarity with dependency grammars helps understand the principle of Dependency Length Minimization (henceforth DLM). It states that natural languages prefer shorter dependencies in their sentences. An example from Hunter and Prideaux (1983), shown in (6) illustrates this.

- (6) a. The janitor threw out the rickety and badly scratched chair.
- 
- b. The janitor threw the rickety and badly scratched chair out.
- 

The study has shown that speakers deem sentences similar to (6a) more acceptable than the ones similar to (6b)¹. Proposed explanations for this preference are based on language-processing constraints, which are usually said to be caused by working memory limitations. With longer dependencies, while reading or hearing a sentence, a person has to keep certain words in their working memory for a longer time. The longer the dependency, the harder the retrieval of the needed word from the working memory. Similar effects have been found in other studies, both psycholinguistic ones (Gibson, 1998; King and Just, 1991) and the ones based on corpus research (Dyer, 2023; Gildea and Temperley, 2007, 2010).

The DLM effect has been observed both at the level of usage and at the level of grammar. The former has been explained already – when speakers have a few grammatically correct ways of phrasing a sentence, they are most likely to choose phrasing that will minimize the distance between connected words. As for DLM in grammar, an example is provided by Futrell et al. (2020). They first compared aggregate dependency lengths within natural language sentences to those in sentences created by randomising the word order while keeping the structure of the tree the same. Random-order sentences turned out to have longer overall dependencies than the natural language sentences, which was proof that, assuming grammar allows for multiple structures of a sentence, people choose the ones with shorter dependencies, therefore DLM works at the level of usage. The next step was to see if it works also at the level of grammar – to this end they checked whether the possible grammatical

¹In the study it is actually found that it's not the distance between the verb and the particle that affects the acceptability of sentences like those, but the syntactic complexity of the phrases within that distance. It was shown however, syntactic complexity as a measure of dependency length correlates with many others proposed, intervening words included (Wasow, 2002).

sentences (not necessarily chosen by speakers) minimised dependency lengths compared to sentences generated by the less constrained grammars. This way they found DLM to be influencing the grammar as well.

One of the earlier formulations of rules similar to DLM was made by Behaghel (1930). He proposed two laws of word order:

1. That which belongs together mentally is placed close together.
2. Of two sentence components, the shorter goes before the longer, when possible.

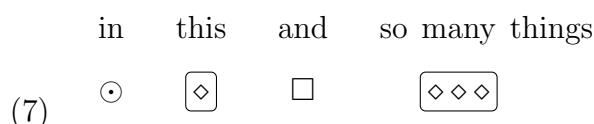
The first of these could be understood as DLM, while the other is a consequence of DLM in head-initial languages. Looking at how syntactic trees are usually shaped in a language allows for a judgement on the headedness or directionality of said language – it can be head-initial if most dependencies are directed to right, or head-final if they are mostly directed to the left. English is an example of a head-initial language, therefore the second law proposed by Behaghel holds for English sentences.

2.3 Possible dependency structures of a coordination

Different corpora choose different approaches to annotating the dependency structure of a coordination. Popel et al. (2013) proposed a taxonomy of those, which consists of three families of annotation styles: Prague, Stanford and Moscow. Przepiórkowski and Woźniak (2023) add to those three a London family. All four of those families are described in more detail in the following subsections. Diagrams are used to better illustrate them, where:

- \odot is the governor of the coordination;
- each \diamond symbolises a token, grouped together with a few others in a rectangle, forming a conjunct;
- \square is the conjunction of the coordination.

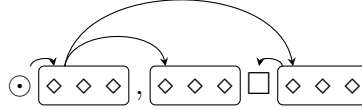
Therefore a coordination presented in (1) using this set of symbols would look like in (7).



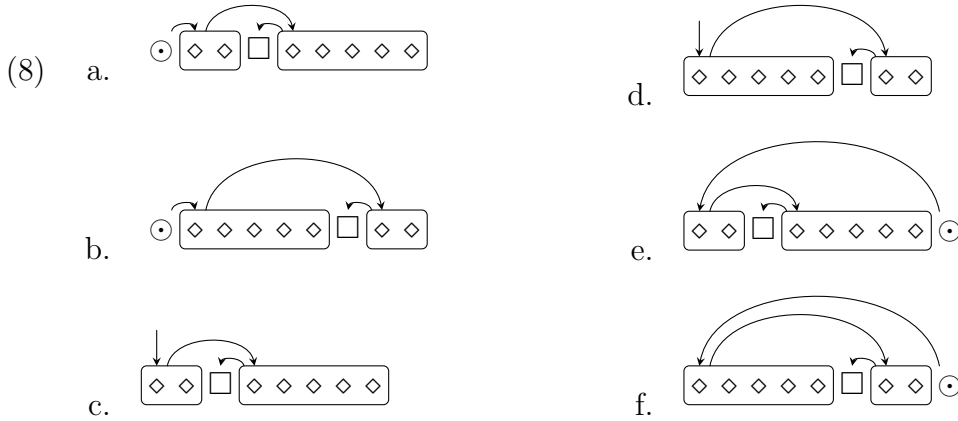
2.3.1 Bouquet/Stanford

The bouquet structure comes from the Stanford parser (de Marneffe et al., 2006). Below is a simplified illustration of such a structure².

²It should be mentioned that in all of the diagrams in Section 2.3 it is assumed that the head of the conjunct is its first word. Such an assumption is justified, because the work presented here is based solely on the English language, which is mostly head-initial, therefore those structures are more likely to be shaped in the way presented here, than in any other.



There is a dependency connecting the governor of the coordination to the first conjunct, which is then connected to the heads of each conjunct, thus forming a bouquet. The conjunction is simply attached to the last conjunct in the structure. This style of annotating is one of the asymmetrical ones, since it does not treat all conjuncts of the coordination equally – it places emphasis on the first conjunct of a coordination by making it the head of every other conjunct.

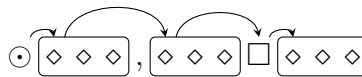


The diagrams in (8) show how the annotation looks depending on different governor positions as well as conjunct lengths. In this particular case, the dependencies are always the shortest when the shorter conjunct is the one on the left, regardless of the position of the governor. This would suggest that, according to the DLM principle, coordinations with shorter conjuncts placed on the left would always be preferred.

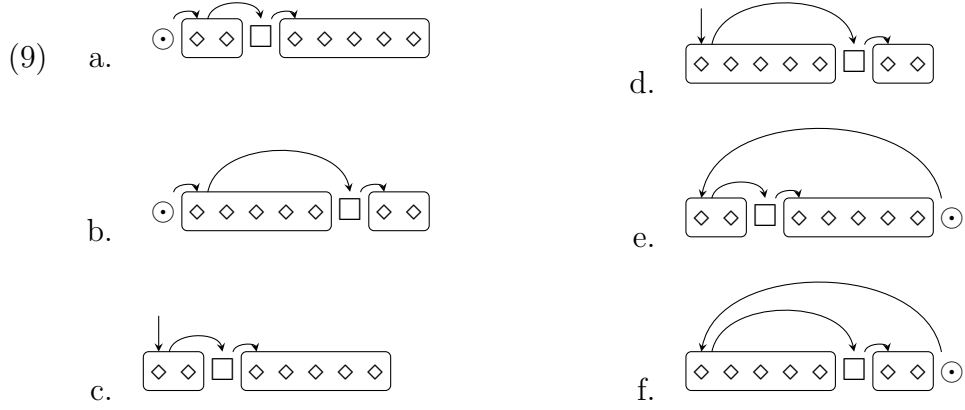
As the Universal Dependencies annotation scheme (described in Section 2.5) was based on the annotations produced by the Stanford Parser, the scheme now uses the Bouquet style to annotate coordinations.

2.3.2 Chain/Moscow

Another asymmetrical approach is the chain, or Moscow one. It is utilised in the Meaning-Text Theory proposed by Mel'čuk (1988).

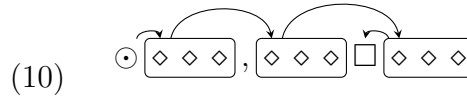


The structure is created by connecting the governor to the first conjunct of the coordination, then every other element of the coordination (including the conjunction) to the next one. As the dependency between the governor and the coordination is specifically between the governor and the first conjunct, the chain approach is another example of an asymmetrical annotation style.



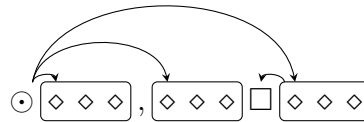
The diagrams in (9) show that in this case, similarly to the bouquet approach, the placement of the shorter conjunct on the left would be preferred, assuming the DLM principle. The placement of the governor again does not seem to have an effect on the ordering.

A variation of this annotation style is now used in the Surface-syntactic Universal Dependencies scheme (described in Section 2.6), which is based on the Universal Dependencies. The authors chose the Chain style instead of the Bouquet, as it minimizes the dependency lengths, which the authors wanted to be reflected in the syntactic structure. The difference between their style and the one shown in (9) is that the conjunction is not attached to the preceding conjunct, but to the next one. This annotation is illustrated in (10).

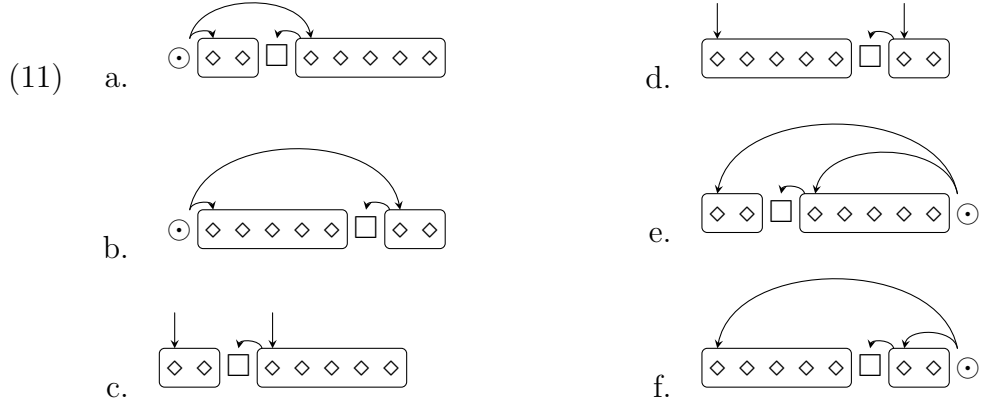


2.3.3 Multi-headed/London

The symmetrical approaches, as the name suggests, treat every conjunct in the coordination the same way. One of them is the multi-headed, or London approach, for which Przepiórkowski and Woźniak (2023) propose the name based on its appearance in Word Grammar developed by Hudson Richard (2010) at University College London.



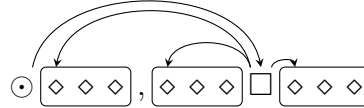
The symmetry of the approach comes from the fact that no conjunct is highlighted by being the only direct dependent of coordinations governor. Instead, all of the conjuncts have dependencies connecting them to the governor, and the conjunction is dependent on the last conjunct, similarly to the bouquet approach.



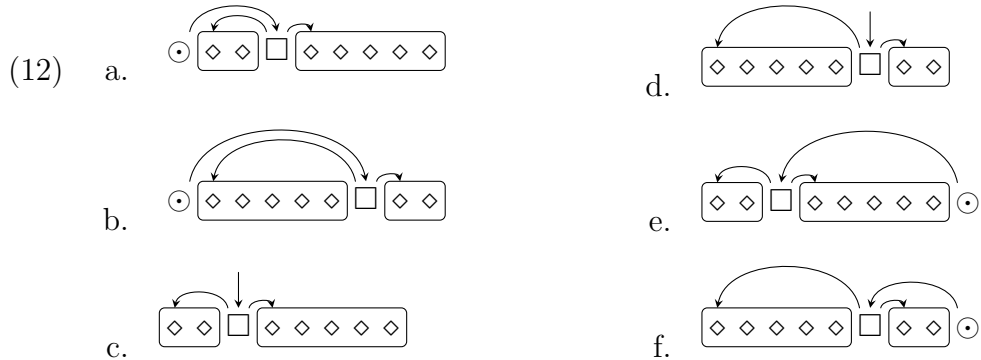
Here, the predictions for the preferred ordering of conjuncts are different to those in asymmetrical approaches. This annotation suggests that the placement of the governor influences the conjunct ordering, specifically that the shorter conjunct will always be placed near the governor – with the governor on the left, as in (11a-b), the dependencies are shorter with the shorter conjunct placed on the left, and with the governor on the right (11e-f) the shorter conjunct should be placed on the right. In case of no word governing the coordination, there seems to be no preference for any ordering.

2.3.4 Conjunction-headed/Prague

The last approach discussed here is the one associated with the Prague Dependency Treebank, called the Prague approach by Popel et al. (2013) or the conjunction-headed by Przepiórkowski and Woźniak (2023).



This is another example of the symmetrical styles, as here again the governor treats all of the conjuncts the same way – in this case, does not connect to any of them. Instead, there is a dependency connecting the governor and the conjunction, which then has the conjuncts of the coordination as its dependents. In case of a coordination without a conjunction, the governor would connect to a punctuation mark.



This annotation style again generates new predictions about the preferred ordering of conjuncts. With the governor placed on the left and without any governor present, this approach should prefer to have the shorter conjunct on the left side of the coordination. With the governor on the right, this approach predicts that ordering does not matter – in both cases presented here the sum of dependency lengths is the same.

2.4 Previous studies

The current study is a replication of Przepiórkowski and Woźniak (2023), who researched coordinate structures to find out whether ordering of conjuncts in English is as simple as placing shorter conjuncts on the left or the placement of the governor of the coordination has some influence on the ordering. They used the Penn Treebank, which is an annotated corpus of texts from the Wall Street Journal. This relatively small, but high quality dataset allowed them to make an argument for the symmetric styles of annotating coordination – they found that the proportion of shorter left conjuncts rose with increasing length differences between conjuncts, but only in coordinations with the governor on the left or without a governor. If the governor was on the right of the coordination, chances for the shorter conjuncts appearing on either side were equal. This is compatible with what the Prague approach would predict, assuming DLM working at the level of usage. The London approach could also explain those results, but only if DLM at the level of grammar was assumed. Since English is a head-initial language, DLM still suggests that shorter dependencies will be placed first, according to the second word of law order proposed by Behaghel (1930). This means that there is a grammaticalised pressure to place shorter conjuncts on the left, and since in case where there is no governor there are no immediate pressures to order the conjuncts in a certain way, the general, grammaticalised pressure may be visible.

Przepiórkowski et al. (2023) have already conducted a replication study of this research. The aims were to see whether the conclusions drawn in the original study hold up when the data come from a bigger and more diverse corpus, but annotated automatically using the Stanza parser. The results were slightly different, but sharpened the conclusions from the original study. In the bigger dataset they found the coordinations with the governor on the left and without a governor to behave the same – with growing length differences between conjuncts the proportion of shorter left conjuncts was higher. Only this time, when the governor was on the right, proportion of shorter left conjuncts was lower with bigger length differences, meaning that the shorter conjunct was drawn to the governor. The Prague annotation predicts that with governor on the right there should be no pressures to order the conjuncts in any way, therefore it turned out to be less compatible than the London approach. Assuming this annotation style, the coordinations with the governor on the right should in fact have shorter conjuncts on the right and the rising proportion of the shorter left conjuncts when there is no governor could be again explained by the grammaticalised pressures of DLM.

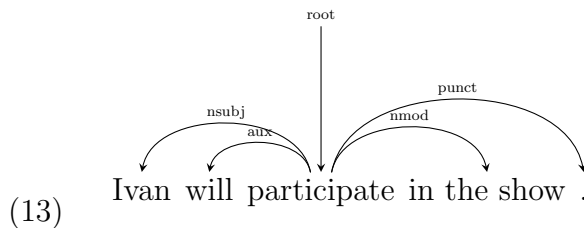
The main issue with this replication study was the low evaluation score of the extracted data. Only 50.1% coordinations in the evaluation sample turned

out to have correct information for analysis. Reasons for such a low outcome, as well as potential solutions are discussed in Section 2.6.

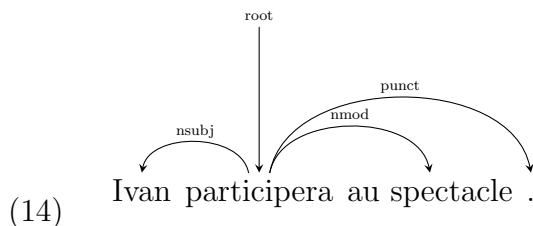
2.5 Universal Dependencies

As seen in Section 2.3, there have been many ideas on how to create dependency annotation. Universal Dependencies (UD, henceforth) is a project focused on formulating guidelines for creating dependency annotation, that would suit as many languages as possible, while maintaining the possibility to represent phenomena specific to any given language. The guidelines outline how one should deal with word segmentation, part-of-speech tagging, assigning morphological features and creating an appropriate dependency structure for a sentence.

Rules for creating a dependency structure are the most relevant part here. In the first version of UD (Nivre et al., 2016) three of them are specified: dependency relations appear between content words, function words are attached to the content words, which they describe, and punctuation marks are attached to the head of the phrase or clause in which it appears. Content words chosen here for dependency heads can otherwise be called "lexical" or "semantic" centers, whereas function words serve mostly a syntactic purpose in a sentence. In the sentence (13) among the words *will* and *participate*, the latter is the one that carries the meaning, thus it is the content word, head of the dependency and, in this case, root of the whole sentence.



The reasoning behind setting those criteria is that it increases the chance of finding similar tree structures in different languages, for example when comparing sentences between English and French, which is morphologically richer. In (14) there is a tree for the French translation of the sentence in (13). Even though the French sentence does not have an auxiliary word to mark the future tense, the structures of those sentences are almost identical, which would not be possible if UD chose to create dependencies between functional words, rather than content words.



For an annotation project on such a significant scale, a unified format for the data was needed, therefore the CoNLL-X format was adapted to the UD needs. This version of the format is used in the current work.

2.6 Surface-syntactic Universal Dependencies

Surface-syntactic Universal Dependencies (SUD, henceforth) is another example of an attempt at creating a set of universal guidelines for dependency annotation. Gerdes et al. (2018) describe it as "near-isomorphic to UD" and offer a set of conversion rules between the schemes. Most of the SUD annotation guidelines are the same as in UD, the most prominent difference is the change in choosing dependency heads, from content words to function words. This section covers the relevant differences between the schemes and how those are beneficial for research described in this work.

2.6.1 Criteria for choosing heads of dependencies

Instead of favoring content words, SUD uses the distributional criteria for choosing dependency heads, which means that "the surface syntactic head determines the distribution of the unit" (Gerdes et al., 2018). It can be tested by checking which of the words within a dependency behaves in sentences similarly to the way the whole unit does, so which of the words can be replaced by the unit and *vice versa*. The example sentence the authors use to explain their criteria is *The little boy talked to Mary*. There is a dependency between words *little* and *boy*, and sentences in (15) and (16) show why the head of this dependency is the word *boy*. In (15a) the word *boy* can be replaced by the unit *little boy*, as shown in (15b), and the sentence is still grammatical.

- (15) a. I saw a **boy**.
b. I saw a **little boy**.

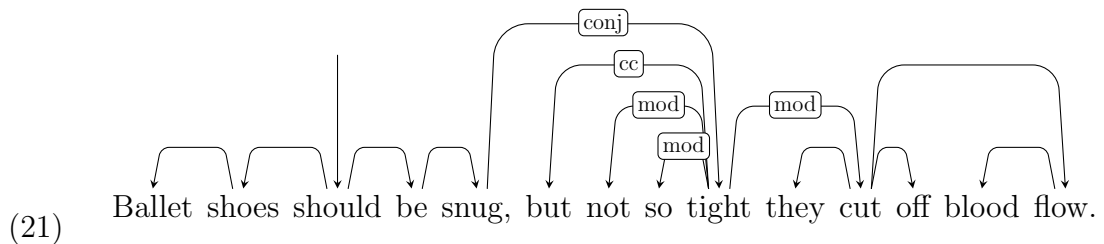
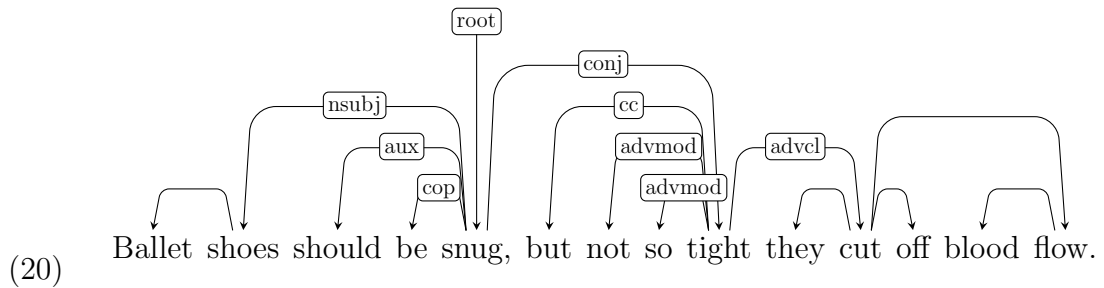
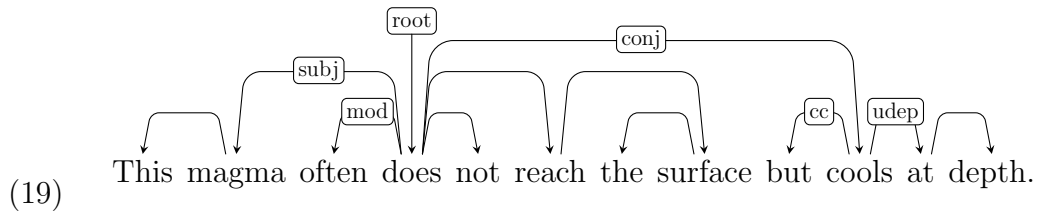
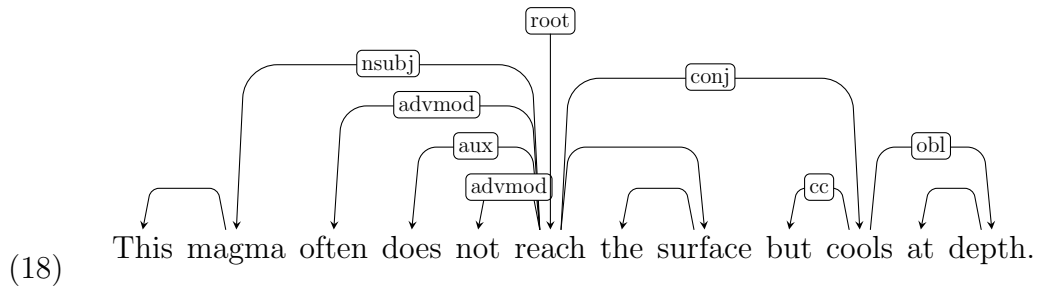
The same is not the case for the word *little*. Trying to replace that word with the whole unit in (16a) results in an ungrammatical sentence in (16b). Sentences (16c-d) shows that replacement in the other way is not possible either, therefore the word *little* cannot be the head of this dependency.

- (16) a. The boy was **little**.
b. *The boy was **little boy**.
c. I found the **little boy**.
d. *I found the **little**.

It is not always possible to test both of the words within a dependency, but in such cases showing that one of the words does not commute with the whole unit is enough to decide it is not the head, therefore the other one must be. As shown in Gerdes et al. (2018), that is exactly the case with the words *to* *Mary* – it is impossible to see how the word *to* behaves on its own, as it needs a noun or a verb, but the sentences in (17) show that *Mary* does not have the same distribution as those two words together.

- (17) a. I saw **Mary**.
b. *I saw **to Mary**.
c. I talked **to Mary**.
d. *I talked **Mary**.

This key difference between UD and SUD has crucial consequences for extracting the exact length of conjuncts in a coordinate structure, which is an integral part of this work. As Przepiórkowski and Woźniak (2023) mention, the UD scheme is not ideal for coordination analysis, as it is not clear which dependencies are shared by the conjuncts and which are private. This is clearly illustrated by the example sentence *Never drink and drive*, where a human parser knows that the word *never* applies to both conjuncts of a coordination, but this would not be obvious to an algorithm. As shown in Przepiórkowski et al. (2023), some heuristics can be employed to find the extents of conjuncts, but they can fail in some cases. It might however be easier to construct accurate heuristics when working with a more syntax-focused annotation scheme than UD, such as SUD.



Based on heuristics used by Przepiórkowski et al. (2023), the tree in (18) would give a coordination with conjuncts *go near my father* and *kept my hand in my pocket*. That would be, because the head of the left conjunct, *go*, has on its left side a dependency labeled *aux* and the head of the right conjunct, *kept* does not have a dependency like this one, therefore *aux* has to be shared by both conjuncts. The correct parse of this sentence gives a coordination with conjuncts *did not go near my father* and *kept my hand in my pocket*, so the dependencies *did not* should not be shared, but private to the first conjunct.

Modifying the heuristics, so that they fit this example, for instance by saying that **aux** dependencies should always be included in the left conjunct, would on the other hand mean that sentences such as in (20) would have incorrect extracted coordinations. In this example the correct coordinate structure has conjuncts *snug* and *not so tight they cut off blood flow*, but was the algorithm to include the **aux** dependency in the first conjunct (as would be required in (18)), the result would be conjuncts *should be snug* and *not so tight they cut off blood flow*.

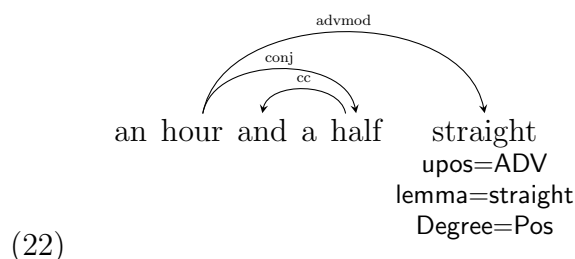
This however is not an issue when using the SUD scheme. There, the words *did not* cannot be dependencies of the coordination, because it is the coordination that is dependent on the word *did*. This way it is ensured that words *did not* will not be a part of the first conjunct and this results in the correct extraction of the coordinate structure. Changing the annotation scheme to SUD does not affect the sentence in (20) – the word *snug* has to be the whole left conjunct, because it also does not have any dependencies in this annotation scheme.

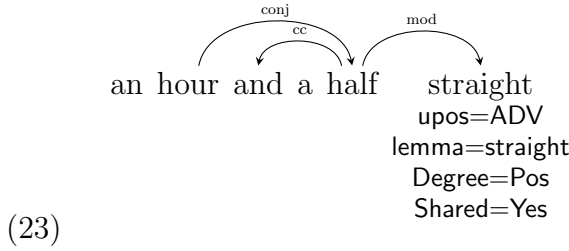
2.6.2 Explicit information about shared dependencies

Besides the structural advantages that SUD has over UD when it comes to analysing coordination, there is one additional feature that is added in SUD treebanks that helps find the extent of conjuncts.

While UD corpora are often created specifically for the purpose of participating in the UD project or converted with manual corrections from different dependency annotations, the SUD corpora are mostly converted automatically from UD. There are a few French treebanks, as well as for Beja, Zaar, Chinese and Naija, that are natively made for SUD, but all others are converted from UD using rule-based graph transformation grammars, which are described in more detail in Chapter 3.

As was mentioned in Section 2.3, UD uses the Bouquet approach to annotating coordination, while SUD uses the Chain one. This means that in the conversion process, some information about the privacy status of a dependency of the coordination can be lost. This is visible in the coordination presented in the sentence *I just sat in there for like an hour and a half straight and studied* (GUM_vlog_studying-27 from the GUM corpus). As annotation in UD in (22) shows, the word *straight* is shared by the whole structure. This is not structurally visible in the SUD version in (23), where the **mod** dependency for the word *straight* is attached to the last conjunct.





So as not to lose this information while converting the annotation scheme, feature **Shared=Yes** is added. Similarly, in coordinations where a dependent is attached to the right conjunct in the UD scheme (therefore private to the right conjunct), during the conversion to SUD the feature **Shared=No** is added.

While the work described here is not based on the conversion code itself, this information still might prove itself useful. The parser was trained on SUD corpora, therefore it might be able to provide additional clues as to which dependents are shared and which are private.

2.6.3 Learnability of dependency schemes

The current study is another replication of Przepiórkowski and Woźniak (2023). As was pointed out in Chapter 1, Przepiórkowski et al. (2023) have conducted identical analysis on the COCA corpus annotated automatically in the UD scheme. After evaluating the automatically annotated data they found only 50.1% of the coordinations in the evaluation sample to be correctly extracted from the corpus. Reason for such an outcome can be twofold: the issues lie either within the parsing accuracy or the script for extracting coordinations from dependency trees.

The latter has been addressed to some extent in Section 2.6.1 – heuristics for finding conjunct extents are easier to develop within a function-word focused annotation scheme. As for the former, there are studies showing that the UD scheme is harder to parse. Rehbein et al. (2017) show that choosing content words rather than function words for dependency heads increases arc direction entropy (measure describing how consistent are dependency directions in a given treebank), which then lowers parsing accuracy. In another study, Kohita et al. (2017) converted UD trees into ones with function heads, rather than content heads. They then used the converted trees for training parsers and parsed 19 treebanks using both UD and converted models. After parsing, the results from the converted model were converted back to UD and for most of the languages (11 out of 19) those results had better scores.

Criterion for choosing dependency heads may not be the only structural advantage that SUD has over UD in terms of parsing. As Gerdes et al. (2018) demonstrate in their article, the Chain approach to annotating coordination, that is used in SUD, minimises the dependency lengths compared to the Bouquet approach used in UD. This may be beneficial for parsing accuracy, as parsers tend to perform better when working with shorter dependencies (Nilsen et al., 2006; Eisner and Smith, 2005).

In the studies cited above the comparison was between UD and a scheme that differed from UD only in some particular aspect, not a new, comprehensive scheme. Tuora et al. (2021) however compared UD to SUD, which matters

because, as they say, "any realistic annotation schema which employs a more 'syntactic' approach to headedness than UD will also differ from UD in the repertoire and distribution of dependency labels, and will also take into account the intrinsic linguistic interaction between various constructions". They trained five parsers, two of which were transition-based and three graph-based, using 21 corpora representing 18 languages. While transition-based parser seemed to perform similarly on both annotation schemes, the graph-based ones preferred SUD. As for attachment scores for the English corpus tested in this experiment (GUM), all of the parsers scored higher with the SUD annotation. The parser utilised in the current study, Stanza, is graph-based and the language of the texts it annotates is English, therefore SUD might be the better choice for the annotation scheme for this data.

Chapter 3

Data processing

The data used in this work is based on the Corpus of Contemporary American English. The corpus consists of raw texts collected in a span of 30 years (1990 – 2019) representing 8 styles: academic, fiction, newspapers, magazines, TV/movies, websites, blogs and spoken. For the analysis of coordinations to be possible, first the texts have to be annotated syntactically. The Stanza parser chosen for this task by default annotates in the Universal Dependencies scheme, but as was explained earlier, the Surface-syntactic Universal Dependencies scheme is preferred here. The parser therefore had to be trained to annotate in the SUD scheme.

The current chapter has two sections: the first one describes the process of training the parsing models that created the syntactic annotation. The second one describes the procedure of finding coordinate structures in parsed sentences and creating a table with data ready for analysis.

3.1 Parser training

Training was conducted using scripts available on github.¹ For the parser to learn annotation, there needs to be already annotated data. The Surface-syntactic Universal Dependencies is a much smaller project than Universal Dependencies, therefore there are not many corpora annotated natively in this scheme. There is however a conversion code that makes it possible to transform any UD data to SUD.²

HERE WILL BE A DESCRIPTION (NOT SURE HOW DETAILED) OF THE CONVERSION CODE

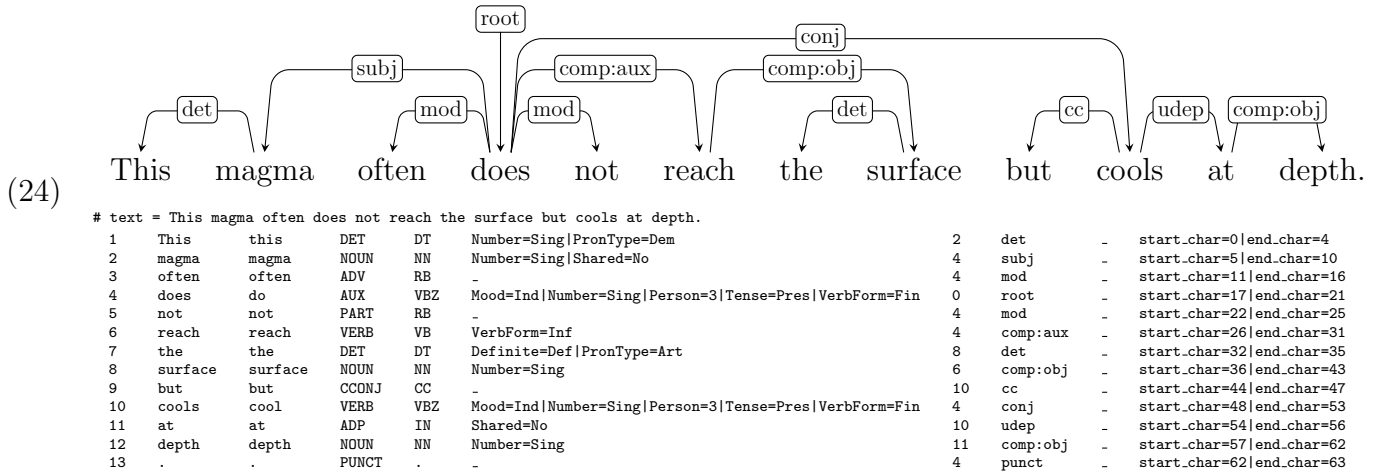
Dependency trees as shown in previous chapters, though possibly comprehensible for people, are not written in a way that is easily understandable by computer programs. Hence, Buchholz and Marsi (2006) created the CoNLL-X format. The exact purpose was to compare parser outputs in a dependency parsing shared task. Today it is widely used for representing dependency trees in a plain-text form. The UD project adapted the format for their needs by replacing some of the information included in CoNLL-X. The adapted version

¹<https://github.com/stanfordnlp/stanza-train>

²https://github.com/surfacesyntacticud/tools/blob/v2.12/converter/grs/UD_to_SUD.

grs

is called CoNLL-U and in (24) there is an example of a SUD dependency tree presented as a tree and in the CoNLL-U format.



3.2 Data extraction

Bibliography

- Behaghel, O. (1930). Zur wortstellung des deutschen. *Language*, 6(4):29–33.
- Buchholz, S. and Marsi, E. (2006). CoNLL-X shared task on multilingual dependency parsing. In Màrquez, L. and Klein, D., editors, *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City. Association for Computational Linguistics.
- Davies, M. (2008-2023). The corpus of contemporary american english (COCA). <https://www.english-corpora.org/coca/>.
- de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In Calzolari, N., Choukri, K., Gangemi, A., Maegaard, B., Mariani, J., Odijk, J., and Tapias, D., editors, *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC’06)*, Genoa, Italy. European Language Resources Association (ELRA).
- Dyer, A. (2023). Revisiting dependency length and intervener complexity minimisation on a parallel corpus in 35 languages. In *Proceedings of the 5th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 110–119.
- Eisner, J. and Smith, N. A. (2005). Parsing with soft and hard constraints on dependency length. In Bunt, H. and Malouf, R., editors, *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 30–41, Vancouver, British Columbia. Association for Computational Linguistics.
- Futrell, R., Levy, R., and Gibson, E. (2020). Dependency locality as an explanatory principle for word order. *Language*, 96:371–412.
- Gerdes, K., Guillaume, B., Kahane, S., and Perrier, G. (2018). SUD or surface-syntactic Universal Dependencies: An annotation scheme near-isomorphic to UD. In de Marneffe, M.-C., Lynn, T., and Schuster, S., editors, *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 66–74, Brussels, Belgium. Association for Computational Linguistics.
- Gibson, E. (1998). Linguistic complexity: locality of syntactic dependencies. *Cognition*, 68(1):1–76.
- Gildea, D. and Temperley, D. (2007). Optimizing grammars for minimum dependency length. In Zaenen, A. and van den Bosch, A., editors, *Proceedings*

of the 45th Annual Meeting of the Association of Computational Linguistics, pages 184–191, Prague, Czech Republic. Association for Computational Linguistics.

Gildea, D. and Temperley, D. (2010). Do grammars minimize dependency length? *Cognitive Science*, 34(2):286–310.

Hunter, P. J. and Prideaux, G. D. (1983). Empirical constraints on the verb-particle construction in english. *Journal of the Atlantic Provinces Linguistic Association*.

King, J. and Just, M. A. (1991). Individual differences in syntactic processing: The role of working memory. *Journal of Memory and Language*, 30(5):580–602.

Kohita, R., Noji, H., and Matsumoto, Y. (2017). Multilingual back-and-forth conversion between content and function head for easy dependency parsing. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 1–7, Valencia, Spain. Association for Computational Linguistics.

Mel'čuk, I. (1988). *Dependency Syntax: Theory and Practice*. The SUNY Press, Albany, NY.

Nilsson, J., Nivre, J., and Hall, J. (2006). Graph transformations in data-driven dependency parsing. In Calzolari, N., Cardie, C., and Isabelle, P., editors, *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 257–264, Sydney, Australia. Association for Computational Linguistics.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal Dependencies v1: A multilingual treebank collection. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Popel, M., Mareček, D., Štěpánek, J., Zeman, D., and Žabokrtský, Z. (2013). Coordination structures in dependency treebanks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 517–527.

Przepiórkowski, A., Borysiak, M., and Głowacki, A. (2023). An argument for symmetric coordination from dependency length minimization: A replication study. Unpublished.

- Przepiórkowski, A. and Woźniak, M. (2023). Conjunct lengths in English, Dependency Length Minimization, and dependency structure of coordination. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15494–15512, Toronto, Canada. Association for Computational Linguistics.
- Qi, P., Zhang, Y., Zhang, Y., Bolton, J., and Manning, C. D. (2020). Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.
- Rehbein, I., Steen, J., Do, B.-N., and Frank, A. (2017). Universal Dependencies are hard to parse – or are they? In Montemagni, S. and Nivre, J., editors, *Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017)*, pages 218–228, Pisa, Italy. Linköping University Electronic Press.
- Richard, H. (2010). *An Introduction to Word Grammar*. Cambridge Textbooks in Linguistics. Cambridge University Press.
- Tesnière, L. (2015). *Elements of Structural Syntax*. John Benjamins.
- Tuora, R., Przepiórkowski, A., and Leczkowski, A. (2021). Comparing learnability of two dependency schemes: ‘semantic’ (UD) and ‘syntactic’ (SUD). In *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics.
- Wasow, T. (2002). *Postverbal behavior*. CSLI Stanford.