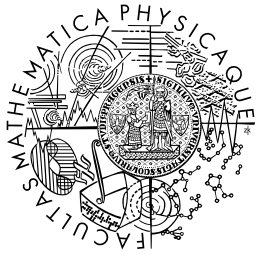


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Ján Faryad

Extraction of multilingual valency frames from dependency treebanks

Institute of Formal and Applied Linguistics

Supervisor of the master thesis: RNDr. Daniel Zeman, Ph.D.
Study programme: Computer Science
Study branch: IML

Prague 2024

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

I would like to express my gratitude to Dr. Zeman for his patience and valuable advice during the work on this thesis. I am also thankful to my parents and brother for their support.

Title: Extraction of multilingual valency frames from dependency treebanks

Author: Ján Faryad

Institute: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Daniel Zeman, Ph.D., Institute of Formal and Applied Linguistics

Abstract: Multilingual valency dictionaries provide helpful information about correspondence of valency frames (verbs and their arguments) across various languages. This work aims at developing a program that automatically creates a multilingual valency dictionary, based on parallel treebanks annotated according to Universal Dependencies. This task includes monolingual extraction of valency frames and their cross-lingual linking. Various methods for solving the task are analysed and implemented. The work includes both general, language-independent approach and additional language-specific extensions, provided in particular for English, Czech and Slovak. The methods for linking the valency frames include using word alignment, morphological and syntactic information contained in the UD annotation or similarity of verbs between related languages. The quality of the solution is evaluated by multiple established metrics on manually annotated data or by comparison with an existing valency dictionary.

Keywords: valency, valency extraction, valency frames, valency dictionary, multilingual extraction, Universal Dependencies, UD

Název práce: Extrakce vícejazyčných valenčních rámců ze závislostních korpusů

Autor: Ján Faryad

Ústav: Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: RNDr. Daniel Zeman, Ph.D., Institute of Formal and Applied Linguistics

Abstrakt: Vícejazyčné valenční slovníky poskytují užitečné informace o shodě valenčních rámců (sloves a jejich argumentů) v různých jazycích. Tato práce se zaměřuje na vývoj programu, který automaticky vytvoří vícejazyčný valenční slovník na základě paralelních korpusů anotovaných podle Universal Dependencies. Tato úloha zahrnuje jednojazyčnou extrakci valenčních rámců a jejich propojení napříč jazyky. Jsou analyzovány a implementovány různé metody řešení. Práce zahrnuje jak obecný, jazykově nezávislý přístup, tak dodatečná, jazykově specifická rozšíření, poskytnutá konkrétně pro angličtinu, češtinu a slovenštinu. Metody propojování valenčních rámců zahrnují použití slovního zarovnání, morfologické a syntaktické informace obsažené v anotaci UD nebo podobnosti sloves mezi příbuznými jazyky. Kvalita řešení je zhodnocena několika zavedenými metrikami na ručně anotovaných datech nebo porovnáním s existujícím valenčním slovníkem.

Klíčová slova: valence, extrakce valence, valenční rámce, valenční slovník, vícejazyčná extrakce, Universal Dependencies, UD

Contents

Introduction

Motivation and aim of the work

Valency is a an important linguistic phenomenon, often classified as deep-syntactic or lying on the border of syntax and semantics. It sees the verb as the center of the simple sentence, which creates slots that are to be filled with arguments of various forms and functions. The verb, as a logical predicate, then expresses the relation among these arguments. The whole of the verb and its valency arguments is called a valency frame. A verb may have multiple valency frames differing in number, form and function of the arguments or in the meaning of the verb itself. Although it shows up that a similar view is applicable probably on all full-meaning parts of speech, the valency of verbs is probably the most complex and the most examined, which holds mostly in this work, too.

In order to study valency in a language, a (monolingual) valency dictionary, that lists verbs and its valency frames, becomes a useful tool. Corresponding verbs in different languages (but one verb can be translated to multiple verbs in the other language, of course) may have different valency frames, differing again in number or function of the arguments (and naturally also in the form, as the languages may use distinct morphological categories). For a given valency frame in one language, a multilingual valency dictionary contains corresponding valency frames (i.e. with verbs of the same meaning) in other languages, whereas also the arguments are mapped one on another.

Such dictionary finds its use in numerous linguistic applications working with multiple languages. It helps to examine relation between the languages concerned or it could be useful for students of a foreign language, to whom it shows differences in verb bindings between their native and the studied language. However, creation of such multilingual dictionary is hard and requires a lot of work, so they are missing for many languages. Fortunately, there are already numerous parallel multilingual data, which might help to build such dictionaries automatically. The aim of this work is to develop a program, that, provided a parallel treebank (but also other scenarios are examined) in any languages, would automatically create a multilingual valency dictionary.

In order to achieve language-independence and best coverage of the program, we need to use treebanks annotated in a universal and widespread way. Universal Dependencies (UD) is a framework that offers probably the most well-established and respected annotation style in dependency linguistics and big, still growing amount of treebanks annotated accordingly, including parallel treebanks. It seems exactly suitable for the task of automatic creation of multilingual valency dictionaries.

According to the assignment of this thesis, the aims can be precisely formu-

lated as follows:

1. Examining the possibilities of multilingual valency frames extraction, including the mapping between corresponding verbs and their arguments.

2. Distinguishing the approaches that are independent of languages or treebanks from language- or treebank-specific methods.

3. Evaluating the quality of the solution on manually annotated data, in particular also the contribution of language-specific approaches compared to the general ones.

In the conclusion of the work, the fulfillment of these aims is discussed.

Structure of the work

The work is divided into four chapters. Chapter 1 treats necessary backgrounds for the work, including valency theory, the data and tools used by the program and previous related works. Particular attention is paid to the UD project and its aspects relevant for valency. The task of the work - the extraction of multilingual valency frames - is composed of two main parts: monolingual extraction of the frames, elaborated in chapter 2, and their cross-lingual linking, to which chapter 3 is dedicated. Both these main chapters contain analysis of its task with a few implementation remarks, description of evaluation process and results and experiments with several possible applications of the program. Various useful technical details are described in the appendices B - D at the end of the work.

Each chapter is divided into several main sections consisting of multiple short subsections, that often contain only one or a few paragraphs. Such fine granularity allows more precise cross-references and hopefully also helps in orientation in the text. Other references in the text lead to tables, figures and also example sentences in various languages, mostly in English or Czech.

Chapter 1

Background

1.1 Universal Dependencies

1.1.1 General introduction of UD

Universal Dependencies (UD) is a project with a goal of creating a cross-linguistically consistent morphosyntactic annotation of human languages and of building a set of such annotated treebanks. This framework is useful for linguistic research and practical language processing across many languages. It focuses on simple representation that would allow analogical processing approach among different languages.[1] Started ca. in year 2014,[2] UD has built over 200 corpora in more than 100 languages,[3] although they differ significantly in the size (ranging from hundreds to millions of words).[4] As the projects proceeds, it improves in more balanced choice of languages and text genres.[5] Over time, UD has actualy become the most widespread and standard dependency annotation scheme, although many proposals on its modification occur.[6][7][8] The principles of UD are based on older attempts such as Standford Dependencies,[9] Google Universal Tagset[10] and Interset[11] which were themselves rooted in several theoretical backgrounds (some of them are shortly presented in the following section 1.2). However, UD is not an a priori linguistic theory, which was later used for a treebank annotation, but rather vice versa, it developed and refined its theoretical principles during the work on the treebanks.

[1] de Marneffe, Manning, et al. 2021, p. 2542
[2] Nivre et al. 2020, p. 4034
[3] *UD web* 2023, `https://universaldependencies.org/`
[4] Nivre et al. 2020, p. 4040
[5] Nivre et al. 2020, pp. 4040–4041
[6] Croft et al. 2017
[7] Przepiórkowski and Patejuk 2018
[8] Gerdes et al. 2019
[9] de Marneffe, Dozat, et al. 2014
[10] Petrov et al. 2012
[11] Zeman 2008

1.1.2 Principles of UD

UD implements only one layer[12] of linguistic annotation consisting of lemmatization and morphological and syntactic description that will be elaborated in the following paragraphs. The decisions for various annotation issues are based on several criteria: UD should be appropriate for description of individual languages, but at the same time sufficiently universal allowing cross-lingual comparison of the annotated phenomena. It should be also comfortable for human annotators, understandable for non-linguists and suitable for various natural language processing tasks. The design of UD tries to balance all these requirements.[13] That means that sometimes one principle must give way in favor of another. For example, despite that UD aims at creating a unified standard for description of all languages, this is not observed absolutely and the traditional annotation customs for individual languages often lead to different annotation conventions for these languages in UD, too, because people using the language are used to them.[14]

1.1.3 Tokenization in UD

Basic units being described and entering into relations are syntactic words, which apart from usual lexical words include also punctuation and other symbols. Each syntactic word is provided with all types of annotation. Every sentence in UD is tokenized into a sequence of syntactic words, which basically correspond to words, but also punctuation marks or other symbols form separate tokens. However, it is more about words as syntactic units than phonological or orthographic ones.[15] This shows to be the simplest solution despite all problems regarding clitics, compound words and other issues emerging from unclear delimitation of word borders among different languages.[16] There are several cases, when multiple syntactic words correspond to a single orthographic word (so called multi-word tokens, e.g. Spanish word *dámelo* = *give me it* consisting of a verb and two pronominal clitics is tokenized into three separate tokens: *da me lo*).[17] Each token is provided with a lemma - a canonical form of the lexeme among all its forms (the choice is based on a language-specific custom; in agglutinative language, lemmas are the forms without any inflectional affix).[18]

1.1.4 Morphology in UD - universal POS tags

The morphological annotation of each token consists of a universal part-of-speech tag – *upostag* and any number of morphological features. The set of possible upostags includes 17 labels (see Tab. 1.1)[19] and cannot be extended.[20] Such set is

[12] de Marneffe, Manning, et al. 2021, p. 268

[13] de Marneffe, Manning, et al. 2021, pp. 302–303

[14] One way to deal with this inconsistency during automatic processing is through language-specific modules that add specific treatments for individual languages to the universal algorithm, which is used also in this work.

[15] Nivre et al. 2020, p. 4035

[16] de Marneffe, Manning, et al. 2021, p. 529

[17] Nivre et al. 2020, pp. 4036–4037

[18] Nivre et al. 2020, p. 4035

[19] *UD web* 2023, `https://universaldependencies.org/u/pos/index.html`

[20] Nivre et al. 2020, p. 4035

Open class words		Closed class words		Other	
NOUN:	noun	PRON:	pronoun	PUNCT:	punctuation
PROPN:	proper noun	DET:	determiner	SYM:	symbol
ADJ:	adjective	NUM:	numeral	X:	other
VERB:	verb	AUX:	auxiliary		
ADV:	adverb	ADP:	adposition		
INTJ:	interjection	PART:	particle		
		CCONJ:	coordinating conjunction		
		SCONJ:	subordinating conjunction		

Table 1.1: Overview of all upostags used in UD.

finer than in traditional theories (e.g. separate class for proper nouns or distinction between coordinating and subordinating conjunctions) and aims on being able to assign each word in each language to one of the parts of speech.[21] There are also technical upostags for punctuation, symbols and other, unclassifiable tokens. The parts of speech play an important role in assigning the dependency relation labels. For this work it is crucial, that upostags capture the difference between full meaning verbs (VERB) and auxiliary verbs (AUX).

1.1.5 Morphology in UD - features

The set of morphological features includes several lexical and inflectional characteristics in an "attribute=value" form. There are 24 universal labels for e.g. reflexivity, pronoun type or foreign words and for many inflectional categories for both nominals and verbs like case, gender, mood, tense or person[22] (see the the overview of all features and their possible values in the cited guidelines).[23] Unlike by upostags, there is a possibility to define other, language-specific features.[24][25] For building a valency frame, the *Case* feature is the most important, but also other features, like *Reflex* (reflexivity), *Voice* or *Person*, are used in this work, especially in the language specific modules.

1.1.6 Syntax in UD - dependency trees

UD is a dependency grammar, that means the tokens are dependent on each other. Dependency is understood as a one-directional binary relation between two tokens: one is dependent on the other.[26]. Each token has its head, that it depends on, and a token can be head for several other tokens. The tokens form a dependency tree structure. A whole subtree of a token is called a phrase, this token is the head of this phrase and other tokens in the phrase depend on it,

[21] de Marneffe, Manning, et al. 2021, pp. 260–261
[22] de Marneffe, Manning, et al. 2021, p. 263
[23] *UD web* 2023, `https://universaldependencies.org/u/feat/index.html`
[24] Nivre et al. 2020, p. 4035
[25] de Marneffe, Manning, et al. 2021, p. 265
[26] de Marneffe, Manning, et al. 2021, p. 257

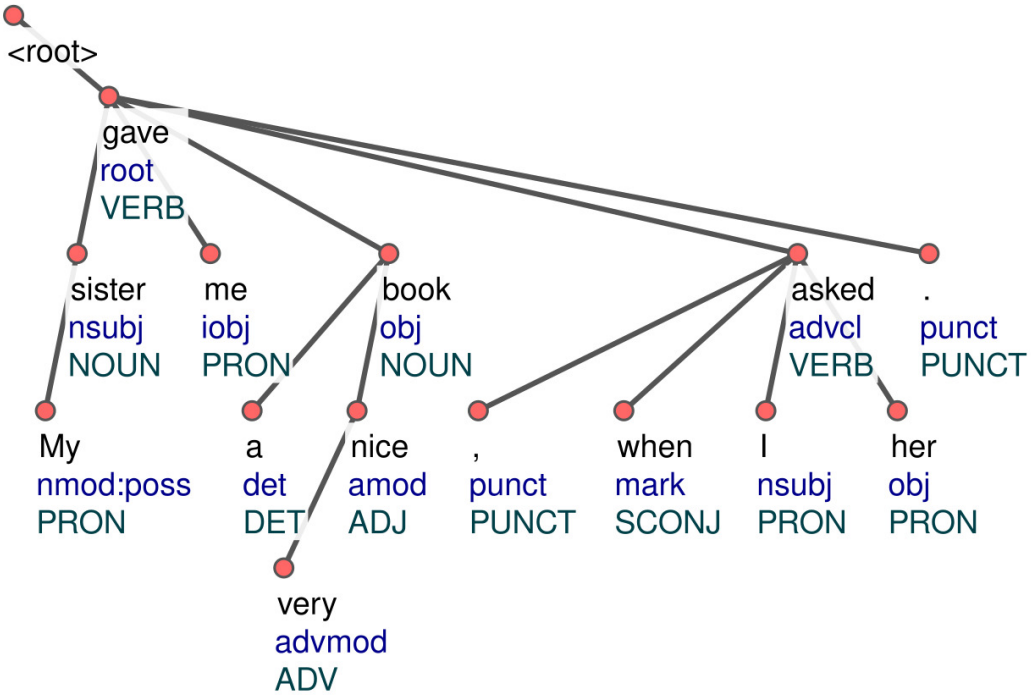


Figure 1.1: Example of a sentence parsed into a dependency tree.

directly (if they are its children) or indirectly (if they are its further descendants). Such a dependency tree can be seen in the Fig. 1.1. UD is oriented on the surface syntax, so only words present in the sentence are participating in syntactic relations (although its extension *enhanced dependencies* offers also option to add artificial nodes for elided predicates[27][28]). Also each sentence is rooted in a technical root which serves as a head of the real root token (so that it holds that each token has its head) and as the representative of the sentence in the treebank.[29]

1.1.7 Syntax in UD - dependency relations

The relation of each token to its head is annotated with a describing label. UD offers 37 basic syntactic dependency relations – *deprels*, with a possibility to define language-specific subrelations for particular deprels[30][31] (see the overview of all the deprels in the cited guidelines or the overview of deprels relevant for valency in the Tab. 1.3 in 1.3).[32] They can be classified into three basic groups: nominal, clausal and modifier dependents. Nominals describe entities that enter into relations with each other. Clauses describe these relations, events and states of various entities that usually depend on the clausal predicate[33] (i.e. the head of the

[27] Nivre et al. 2020, pp. 4039–4040

[28] *UD web* 2023, `https://universaldependencies.org/u/overview/enhanced-syntax.html`

[29] de Marneffe, Manning, et al. 2021, p. 257

[30] Nivre et al. 2020, p. 4036

[31] de Marneffe, Manning, et al. 2021, p. 265

[32] *UD web* 2023, `https://universaldependencies.org/u/dep/index.html`

[33] de Marneffe, Manning, et al. 2021, p. 272

clause), which is usually a verb[34] (but not necessarily, nominal predicates are also very common, see (1)). Modifiers add attributes to nominals, clauses and other modifiers.[35] Moreover, there are also many deprels for various function words and other auxiliary deprels, that do not describe dependency in a proper sense, but serve rather as technical means to capture phenomena like coordination, ellipsis, multi-word expressions or to include e.g. punctuation.[36][37]

(1) Various upostags working as predicates:
"The child plays[VERB] tennis."
"The child is a liar[NOUN]."
"The child is smart[ADJ]."

Besides the division above, the deprels are also divided according to whether they describe a dependency on a predicate or not. The dependents on predicates are further divided into core and non-core, which is a distinction specific for UD. It is highly relevant for valency, the main topic of this thesis, and it is elaborated in more detail in the section 1.3, after introduction into the valency itself. Individual deprels relevant for valency are described there, too.

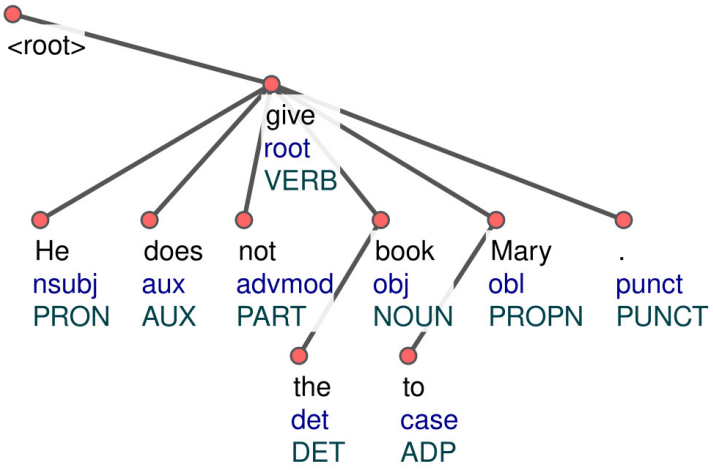
1.1.8 Primacy of content words in UD

One more important syntactic principle holds in UD, that shows to be relevant for our topic. In various languages, there can be an autosemantic word (with full meaning on its own) syntactically driven (i.e. required a specific form) by a synsemantic auxiliary word (not having full meaning) that enters into syntactic relations with other words in the sentence (e.g. prepositional phrases or compound verb forms with auxiliary verbs). Dependency theories differ in annotation of these cases, whether the head of such expression is the synsemantic word (syntactic priority) or the autosemantic word (semantic priority). The UD considers the latter way more universal, because languages differ in the use of syntactic means (auxiliary words, morphological forms, word order etc.), but the semantics of corresponding sentences should remain always similar and thus comparable.[38] This is illustrated on a corresponding sentence in English (2) and Czech (3) with tree structures shown in Fig. 1.2. The same functions (marked by corresponding colour) are expressed by auxiliary words on the English side (Fig. 1.2a) and by morphological means on the Czech side (Fig. 1.2b), but if we neglect them in both sentences, the dependency tree remains equivalent (Fig. 1.2c). The principle of the primacy of autosemantic words (also called content words)[39] is important for this work and I will refer to it at several places.

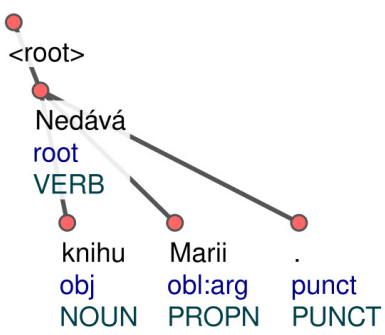
(2) "He[3pers] does[3pers] not[neg] give the book to[ADP] Mary."

(3) "Nedává[3pers, neg] knihu Marii[case:Dat]."

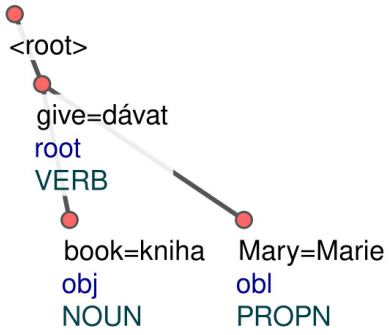
[34] de Marneffe, Manning, et al. 2021, pp. 257, 272
[35] de Marneffe, Manning, et al. 2021, pp. 257–258
[36] Nivre et al. 2020, p. 4036
[37] de Marneffe, Manning, et al. 2021, pp. 276–286
[38] de Marneffe, Manning, et al. 2021, p. 2573
[39] *UD web* 2023, `https://universaldependencies.org/u/overview/syntax.html`



(a) English sentence



(b) Czech sentence.



(c) Common sentence structure

Figure 1.2: Comparison of structures of a corresponding sentence in English and Czech.

1.1.9 Application of UD guidelines

Unless said otherwise, all descriptions and examples present in this work are base on the official UD guidelines[40], which does not need to correspond always with the actual annotation present in the UD treebanks (1.4.3) or the annotation given by the UDPipe (1.4.6). This can be because of the mistakes made during the annotation, but often also because of annotation customs for individual languages (based on a language- or theory-specific tradition), which may still differ from the guidelines.

1.2 Valency theory

Although valency syntax is already an established area of syntax in the present time, there might be still some little differences in the precise meaning of the terminology across different linguistic traditions. After all, the UD project itself is based on different previous linguistic theories. For a better understanding of what the valency actually is, it is necessary to examine its role in several

[40] *UD web* 2023, `https://universaldependencies.org/guidelines.html`

traditions in the past century. We will shortly look at the notion of valency in Lucien Tesnière's work, in the theory of the American Charles Fillmore, the understanding of valency in the Prague linguistic tradition and finally at the connection between valency and UD deprels.

1.2.1 Tesnière's view on valency

Lucien Tesnière can be considered a father of the valency theory. He divided words governed by a verb to two categories: actants and cirmumstants. Actants are entities participating in the action expressed by the verb.[41] They are nouns or at least words that behave like nouns.[42] On the other hand, circumstants describe some circumstances of the action (like time, place, manner, etc.) and they are mostly adverbs or something behaving in that way.[43] Unlike the number of actants (see below), the number of circumstants is not limited.[44] However, the dividing line between actants and circumstants is not always clear.[45] Besides the noun – adverb distinction, there is often another criterion used: whether the function of the word is indispensable for completing the meaning of the verb (this leads to an actant) or not (in that case the word is a circumstant).[46]

Tesnière observes that different verbs can govern different number of actants.[47] According to him, valency is a number of actants a verb can govern.[48] He distinguishes verbs with zero to three actants, which are numbered first, second and third.[49] According the actant number, the verbs are called *avalent*, *monovalent*, *bivalent* and *trivalent*. These actants differ in the way they are perceived by the speakers of the language, so there can exist e.g. monovalent verbs that always govern only the second or the third actant instead of the first one: in the German sentence (4), the actant *mir* is perceived in the same way as *mir* in (5), where it has a role of the third actant (*Du* is the first one and *Buch* is the second one).[50]

(4) "Es ist mir warm." (DE) = lit. "It is to_me warm."

(5) "Du gibst mir ein Buch." (DE) = "You give me a book."

For bivalent verbs, Tesnière distinguishes four voices, which cover different relations of actants to the role they play in the action expressed by the verb: active, passive, reflexive and reciprocal.[51] These can be applied also to the trivalent verb, but the inversion of actants can happen in two ways: between the first and the second or between the first and the third.[52] For example, active sentence (6) can be inverted in the first way to a passive sentence (7) or in the second way to a

[41] Tesnière 1959, chap. 48, §4
[42] Tesnière 1959, chap. 48, §6
[43] Tesnière 1959, chap. 48, §7-8
[44] Tesnière 1959, chap. 56, §4
[45] Tesnière 1959, chap. 57, §1
[46] Tesnière 1959, chap. 57, §4
[47] Tesnière 1959, chap. 50, §4
[48] Tesnière 1959, chap. 97, §3
[49] Tesnière 1959, chap. 51, §3
[50] Tesnière 1959, chap. 99, §9
[51] Tesnière 1959, chap. 100, §5-10
[52] Tesnière 1959, chap. 106, §10

different passive sentence (8).[53]

(6) "Alfred gives the book to Charles."

(7) "The book is given by Alfred to Charles."

(8) "Charles is given a book by Alfred".

Tesnière also notices that trivalent verbs can miss the third actant while having two second actants (9)[54] or vice versa (10).[55] He supposes that there is no language with tetravalent verbs, but he admits that in the future they may arise. His view on development of languages assumes that all actants were historically circumstants, so monovalent verbs developed from the avalent by transformation of one of its circumstants into an actant. The same holds for bivalent ant trivalent verbs and this development is said to continue.[56]

(9) "Antonius pueros*[ACC]* grammaticam*[ACC]* docet." (LA)
= lit. "Antonius children grammar teaches."

(10) "Id est mihi*[DAT]* gaudio*[DAT]*." (LA) = lit. "That is to_me to_joy."

He also examines relations between the verbs that differ only in number of actants.[57] The increase in the number of actants is called causative diathesis and the new actant usually takes the position of the first actant (11), (12).[58] Beside of using a different lexeme, the causative can be formed through the use of an auxiliary verb[59] or through a derivational affix added to the original verb[60] or even with no marker at all.[61] In the latter case both verbs (the original and the causative one) have exactly the same form (13).[62]

(11) die – kill: "Alfred*[1]* dies." – "Bernard*[1]* kills Alfred*[2]*."

(12) see – show: "Alfred*[1]* sees Bernard*[2]*." – "Charles*[1]* shows Bernard*[2]* to Alfred*[3]*."

(13) "Alfred*[1]* lives." - "Alfred*[1]* lives a good life*[2]*."

The reverse process is also mentioned: recessive diathesis means reducing the valency of a verb by one. This is in many languages realized by the same marker as the reflexive voice creating a problem of distinguishing the true meaning of this marker, when used. In the sentence (14), the pronouns *se* does not have its usual reflexive, but recessive meaning.[63] This phenomenon is sometimes viewed as a periphrastic passive. But Tesnière recognizes also the regular passive as a

[53] Tesnière 1959, chap. 106, §12
[54] Tesnière 1959, chap. 106, §13
[55] Tesnière 1959, chap. 106, §16
[56] Tesnière 1959, chap. 106, §20
[57] Tesnière 1959, chap. 107
[58] Tesnière 1959, chap. 107, §3
[59] Tesnière 1959, chap. 112
[60] Tesnière 1959, chap. 113
[61] Tesnière 1959, chap. 114
[62] Tesnière 1959, chap. 114, §7
[63] Tesnière 1959, chap. 115, §9

marker of recessive diathesis, e.g. in a passive sentence (15), the verb has recessive meaning.[64] Finally, also the recessive diathesis can have a zero marker,[65] as in (16), where the usually bivalent verb takes a monovalent function.

(14) "Ce livre se lit facilement." (FR) = lit. "This book itself reads easy.", meaning "This book is easy to read."

(15) "Homes are built quickly in this country."

(16) "The door opens."

1.2.2 Fillmore's view on valency

Another theoretical source, relevant for study of valency, is the work of the American linguist Charles Fillmore. Fillmore was originally a generative linguist (theory established by Noam Chomsky based on constituency or phrase structure syntax instead of dependency syntax), but during his lifetime, he built his own theory, called *Case grammar*, later transformed into *Frame Semantics*. Although not using the word *valency* in his early papers, he actually deals with valency frames and valency arguments of verbs.

According to Fillmore, syntax should stand in the centre of linguistic interest. From this perspective he elaborates on the term *case*, which used to be understood primarily morphologically as different forms of nouns in some languages. However, nouns in different cases enter into semantically relevant syntactic relations with the verb. Fillmore believes that these relations apply universally and that it is possible to describe them across all languages.[66] Individual cases have often several meanings in one language. In grammars for classical languages we can find different use cases of one case, like *dative of possession*, *accusative of time* etc.; a case rarely has just a single meaning.[67] These use cases can vary among different case-based languages, because even if two cases in two compared languages would have the same name characterizing their most common and most frequent meaning, they might differ in other meanings, which would make any cross-lingual conclusions difficult.[68] Fillmore therefore suggests comparing these deep meanings and this is what he actually understands under the term *case*, as he uses it, instead of surface forms.[69]

The transition to a semantic understanding of cases is not the only shift Fillmore makes regarding this term. He observes that the functions, which are in case-based languages represented by cases, are in languages without cases represented by prepositions (e.g. English) or postpositions (e.g. Japanese). Even in languages with cases, the functions are often combined and their representation includes both cases and adpositions (a term for prepositions and postpositions together). Fillmore therefore classifies also adpositional relations under this term.[70] Especially this idea has a significant response in UD annotation.

[64] Tesnière 1959, chap. 116, §3
[65] Tesnière 1959, chap. 117
[66] Fillmore 1968a, pp. 23–25
[67] Fillmore 1968a, p. 29
[68] Fillmore 1968a, p. 26
[69] Fillmore 1968a, pp. 41–42
[70] Fillmore 1968a, p. 36

Fillmore also realizes that the subject of the sentence takes different semantic functions: when a sentence is inverted into the passive voice, the former object becomes subject, but semantically it has the same function as before. Fillmore, inspired by Tesnière, sees no reason to perceive subject as a substantially distinct verb argument from the objects. According to him, subject is only a surface syntactic relation, without any significant importance for semantics.[71] He describes basic sentence structure as a verb with several noun phrases around, that are in some specific relation (case) to the verb. Moreover, every such relation can occur only once (unless coordinated).[72]

These observations led Fillmore to establish a system of several *semantic* cases. The first of them are Agentive (animate source of the action), Instrumental (inanimate instrument), Dative (animate element influenced by the action), Objective (inanimate element influenced by the action), Factitive (result of the action) and Locative (spatial information); later he adds more of them.[73] The information about which cases a verb takes is an integral part of it. Fillmore calls it *case frame*. Each verb can have several case frames describing which cases it requires in the sentence. Some of the cases are optional, some of them form a group from which one case should be chosen and some are completely obligatory, but even then they can be dropped if the context allows it.[74][75] Unlike Tesnière, Fillmore observes the existence of verbs with four arguments: e.g. verbs *sell* and *buy* need a seller, a buyer, goods and money.[76]

1.2.3 Functional Generative Description and its valency representation

Functional Generative Description (FGD) is a complex linguistic theory developed in Prague since the 1960s. It is significantly inspired by works of Tesnière 1.2.1 and Fillmore 1.2.2, but builds also on a strong linguistic legacy of the Prague linguistic circle, active in Prague in the 1930s.

FGD has an extraordinary meaning for this work: it is one of the main sources for UD 1.1 and moreover the existing valency dictionaries (Vallex 1.4.5), used for evaluation of the automatic valency extraction in this work, are built upon the FGD.

The basic concept of FGD is a description of a sentence with a system of five layers (the number was later reduced to four when annotating treebanks according to this theory), where on each layer, the sentence is represented with a structure of elementary units combined into complex units.[77] Complex units of a lower layer (i.e. closer to the surface) correspond to elementary units of another, higher layer (also called a deeper layer). In this correspondence, the unit on the lower layer is called a form of the higher counterpart and the other way around, the unit on the higher layer is called a function of its lower representation. The original idea of the way this layer system works is inspired by generativist tradition: the

[71] Fillmore 1968a, p. 38
[72] Fillmore 1968a, pp. 41–42
[73] Fillmore 1968a, pp. 46–48, 54
[74] Fillmore 1968a, pp. 49–51
[75] Fillmore 1968b, p. 75
[76] Fillmore 1968b, pp. 75, 80
[77] Vernerová 2019, p. 59

aim is to generate correct sentences, where the function on each layer generates its forms for the lower level using a set of rules. This idea is expressed in the two attributes in the name of the theory: functional and generative. In fact, the system can be used quite well also in the opposite direction, i.e. for analysis of given surface sentences.

The five layers are: phonetic, morphophonemic, morphemic, surface-syntactical and tectogrammatical. The first three layers describe the sentence as a linear sequence of their units, whereas the two last layers represent the sentence with a tree structure. The surface-syntactical layer (also called "analytical" in the FGD-based corpora) is important to us, because it captures the words of a sentence as tagmemes (sentence parts) and assigns them traditional surface-syntactical labels such as subject, object, attribute etc. However, it is the deepest, tectogrammatical layer that is used to represent valency.[78]

The valency in FGD is realized by valency frames – sets of slots for complements created by a lexical unit and described with a fuction label (*functor*), form requirements and other information (obligatoriness, repetitiveness etc.).[79] The notion of valency frames roughly corresponds to Fillmore's case frames. The lexical units with valency frames are not only verbs, but also nouns, adjectives and adverbs, but this requirement of the theory is not met in the existing valency dictionaries – only verbs are systematically annotated for valency. Moreover, non-verbs with valency are often derived from a verb: like a verbal noun *odhodlání k+DAT* (*determination to*, *resolve to*) derived from the verb *odhodlat se k+DAT* (*to resolve to*) or verbal adjectives *zbavený +GEN* (*rid of*, as in "A dog rid of fleas can enter the house.") and *sahající po+ACC* (*reaching to*) derived from the verbs *zbavit +GEN* (*to rid of*) and *sahat po+ACC* (*to reach*), respectively.

The lexical units can share one lexeme, but each of the units represents a different meaning of the verb with different valency frame. For example the verb *odpovídat* (*to respond*) has a lexical unit for the meaning *to give an answer* with a valency frame A[NOM] *odpovídá* B[DAT] na_C[ACC],(A *responds* to_B on_C, as in "Odpovídá mi na dotazy", lit. "He_responds to_me on_questions", meaning "He responds to my questions") and then it has another lexical unit for the meaning *be responsible* with a frame A[NOM] *odpovídá* za_B[GEN] (A *is responsible* for_B). An English example can be the verb *to try*, which can mean *to have a try* with a frame A *tries* B (e.g. *a cake*), *to make effort* with an optional verbal object (e.g. "I try hard (to succeed)") or *to judge* with a frame A *tries* B (*a criminal*) for C (*a crime*).

Forms of verbs differing only in aspect (perfectness and imperfectness or imperfectness and iterativeness) are considered the same lexeme and they can be shared by all lexical units of the lexeme, or they may work only for some of the lexical units. For example there is a Czech lexeme *odstoupit* (pf.) / *odstupovat* (impf.) with the two forms with a different aspect and with at least two lexical units differing in meaning (*to step back* vs. *to resign* and valency frame, but both the units share both the forms of the lexeme. But for the lexeme *odpovědět* (pf.) / *odpovídat* (impf.) mentioned above, only the lexical unit for *to give an answer* works with both forms, whereas the unit with the meaning *to be responsible* can

[78] Vernerová 2019, p. 68

[79] Vernerová 2019, pp. 70, 107

be expressed only by the imperfective form.[80]

There are two basic types of complementations: actants (also called inner participants or arguments) and adjuncts (also called free modifications); they correspond to Tesnière's actants and circumstants. The actants uniquely characterize the verb, they can be used only once (members of coordination or apposition are treated as one actant) and they are mostly obligatory (but not necessarily). There were identified five actant functors: actor (ACT, the actual doer of the action), patient (PAT, object influenced by the action), addressee (ADDR, recipient or beneficiary of the action), origo (ORIG, cause or source of the action) and effect (EFF, result of the action). A typical example of a Czech sentence containing all five actants is "Matka.ACT předělala dětem.ADDR loutku.PAT z kašpárka.ORIG na čerta.EFF. Mother.ACT remodeled.ADDR the puppet.PAT for children from a clown.ORIG to a devil.EFF."[81]

On the other hand, the adjuncts can be used multiple times, they are mostly optional and they can be used for almost all verbs. They participate on the verb characteristics only if they are obligatory (in that case they are also a part of the valency frame). There is a large number of functors for them: temporal, spatial and of many other types.[82]

An important part of valency in FGD is a concept of actant shifting. If there is only one actant realized in the sentence, it is always labeled as actor, regardless what its true (semantically most relevant) functor should be. Similarly, if there are two actants, they are always actor and patient. When these two main actants are used, only then can be used also the other three labels. Although actant shifting may introduce a little confusion, there are valid reasons for it. It corresponds to the surface sentence form in most languages, where the majority of monovalent verbs has a subject and a majority of bivalent verbs has a direct object. It is an attempt for a compromise between the pure syntactical Tesnière's approach, who used only numbering of the actants without any labels, and the pure semantic Fillmore's approach, who based his labels only on semantic role of the actants without any formal criteria for assigning them.[83] For illustration, in the sentence "Dítěti.ACT[Dat] lezou zuby.PAT[Nom].", lit. "To_child.ACT they_climb teeth.PAT.", meaning "The child's teeth are coming out", the word marked as the actor is in dative case, in which addressees are commonly expressed. But because of the actant shifting, only actor and patient should be used in this valency frame, because it has only two actants. It is also remarkable, that the subject is marked as patient, although the sentence is not in the passive voice. This shows us that the assignment of the actant labels is still partially semantically motivated and does not happen automatically based on the syntactic structure of the sentence.

The set of accepted forms for a given complementation is either explicitly listed (this holds typically for all actants) or it is clear from the functor. The forms can be direct or prepositional cases, infinitives, dependent clauses or a few other, special constructions or parts of phrasemes.[84]

[80] Vernerová 2019, p. 73
[81] Vernerová 2019, p. 82
[82] Vernerová 2019, p. 86
[83] Vernerová 2019, p. 87-90
[84] Vernerová 2019, p. 105-106

	actants	adjuncts
obligatory		
optional		

Table 1.2: Overview of verb complements constituting the valency frame in FGD (shown in red).

The theory defines valency frame, which is a set of complements that characterize the particular verb. A valency frame is composed of obligatory and optional actants and obligatory adjuncts (typical, but still optional adjuncts may be listed as well, although they are not part of the valency frame), as shown in Tab. 1.2. The idea is to include into the frame all information that cannot be deduced by applying general rules when generating lower layers.[85] The actants, both obligatory and optional, are by definition complements, that are specific for the verb, they cannot be used arbitrarily with any verb, so they must be included into the frame characterizing the verb. The adjuncts can be used with any verb several times, but if a verb enforces them, they are also part of its characteristics.

For the purpose of determining, which complements are obligatory, a set of dialogue tests was invented. An obligatory complement might not be explicitly expressed, but only if it is known from the context. So when the speaker is asked about it, he cannot say, he does not know:

"He is arguing."
"Who is he arguing with?"
"*I do not know."
→ The second participant of the argument (with the preposition *with*) is an obligatory actant (but unexpressed in the original sentence), because the speaker cannot say, he does not know.

"He is arguing."
"What is he arguing about?"
"I do not know."
→ The issue of the argument (with the preposition *about*) is an optional actant, because the speaker does not need to know it.

Similarly we can test not only actants, but also adjuncts:

"They came back."
"(To) where?"
"*I do not know"
→ The final destination of their movement is an obligatory adjunct, because the speaker cannot say they came back without knowing where they came to.

"They came back."
"From where?"
"I do not know"
→ The original destination of their movement is an optional adjunct, because the speaker does not need to know, where they were before.

[85] Vernerová 2019, p. 107

The idea of valency frames presented in FGD is used also in this work. We will use the notions of actants and adjuncts as described in this section, but the definition of a valency frame will be further revised (see 2.2).

1.3 Valency-related dependency relations in UD

UD does not have an elaborated valency theory as the ones above, but its division of deprels (see Tab. 1.3 for relevant deprels, for the full table of all the UD deprels, see the guidelines[86]) into core and non-core dependents (see 1.2) weakly indicates the nature of relations between the verb and its dependents in this regard. The notion of the core is nevertheless different from the actants in the FGD. UD intentionally avoids the distinction between actants and adjuncts[87], arguing that the difference is pretty unclear and frequently argued over[88] and the criteria defined by Tesnière and following theories or various tests often give opposing classification of a given dependent,[89] so the best solution is to eliminate it. The notion of UD core covers a subset of actants, it includes roughly only subject and direct object, leaving other objects out. The dependents having one of the core deprels should be certainly included into the valency frame. Regarding the non-core dependents, they can correspond both to actants and adjuncts, so they should be further examined regarding their inclusion into the frame. In the following paragraphs, individual deprels relevant for valency are described.

	nominals	**clauses**	**other words**
dependents of core predicate	nsubj obj iobj	csubj ccomp xcomp	
dependents of non-core predicate	obl expl	advcl	advmod aux cop mark compound
dependents of non-predicate			case

Table 1.3: Overview of the discussed, valency-related deprels. The columns describe the form of the dependents: whether they are nominals, clauses or modifier and function words. The rows mark, whether they depend on the predicate and if yes, whether they are a part of its core.

[86] *UD web* 2023, `https://universaldependencies.org/u/dep/index.html`

[87] de Marneffe, Manning, et al. 2021, p. 268

[88] *UD web* 2023, `https://universaldependencies.org/workgroups/core.html#avoiding-an-argumentadjunct-distinction`

[89] Przepiórkowski and Patejuk 2018, pp. 3837–3838

1.3.1 *nsubj* - nominal subjuct

UD keeps the traditional distinction between subject and other actants. A nominal subject is labeled with a *nsubj* deprel (17). In some languages, the *nsubj* deprel can be specified to *nsubj:pass*, if it is a subject of a passive construction, indicating that this actant is actually a patient rather than a proper agent of the action, as in "The dinner is being eaten", where the word *dinner* is labeled with the *nsubj:pass* deprel (18). This allows to do a transformation when extracting the valency frames, so that the same meaning of the verb is not extracted with two separate frames differing only in their voice.[90][91]

(17) "Cats*[nsubj]* chase mice."

(18) "Mice*[nsubj:pass]* are chased by cats."

1.3.2 *csubj* - clausal subject

The subject can be replaced with a subordinate clause. This clausal subject, more precisely the head of the whole clause (i.e. the predicate, mostly a verb) receives a *csubj* deprel (19).[92][93] The clausal subject can be used in passive constructions, too, receiving thus a label *csubj:pass* (20).

(19) "Who comes*[csub]]* first is the winner."

(20) "Who comes*[csubj:pass]* first will be rewarded."

1.3.3 *obj* - object, *iobj* - indirect object

Another traditional core argument which is present also in UD annotation is the direct object. It is marked with *obj* deprel and typically represents the patient of the action. Distinguishing of subject and object in English lies mostly in their position in the sentence: the subject comes before the verb, the object follows it (21). However, in many languages, there can be changeable word order (e.g. allowing stressing one of these actants) and the identification lies rather in morphological cases, e.g. in Czech, the objects takes in most cases the accusative case (22). There are also verbs requiring their object in other cases, but they do not get the *obj* deprel in UD (see below).[94][95]

(21) "I see a woman*[obj]*."

(22) "Vidím ženu*[ACC,obj]*." = "I see a woman."
"Ženu*[ACC,obj]* vidím." = "I see a woman."

In some languages, there are verbs with multiple objects in the same, prepositionless case. For example in the sentence (23), both objects marked are such and

[90] de Marneffe, Manning, et al. 2021, pp. 273–274
[91] *UD web* 2023, `https://universaldependencies.org/u/dep/nsubj.html`
[92] de Marneffe, Manning, et al. 2021, pp. 277–279
[93] *UD web* 2023, `https://universaldependencies.org/u/dep/csubj.html`
[94] de Marneffe, Manning, et al. 2021, pp. 273–274
[95] *UD web* 2023, `https://universaldependencies.org/u/dep/obj.html`

similarly in the Czech sentence (24). Beside of this, there can be other secondary objects used with prepositions or in non-typical cases, that are traditionally called indirect. In order to keep consistency of annotation among many languages with different perception and traditional annotation of objects, UD aims on having only one direct object (*obj*) per predicate. If there are multiple such objects (traditionally perceived as direct), they should be labeled as indirect and take an *iobj* deprel, as we can see in both the examples mentioned.[96][97]

(23) "I gave him*[iobj]* a book*[obj]*."

(24) "Učím děti*[iobj]* němčinu*[obj]*."
= "I teach children*[iobj]* German*[obj]*."

The decision, which of the two objects is considered direct, is traditional, with the help of some transformations that show one of the object to be indirect, i.e. in a non-typical form (they would be labeled *obl* in UD - see 1.3.4), e.g. in (25) *him* is not direct because of the preposition, whereas *book* is still direct. Or in the passive voice, there are two options (one transformation for each object): in (26) *him* shows as indirect object, too, whereas in (27) *book* keeps its direct form. This leads to the decision to consider *him* a secondary object and mark it with the *iobj* deprel.

(25) "I give a book*[obj]* to him*[obl]*."

(26) "A book*[nsubj]* is given to him*[obl]*."

(27) "He*[nsubj:pas]]* is given a book*[obj]*."

However, annotation custom for some languages do not abide by the guidelines as described and use the *iobj* deprel for traditional indirect objects, which should be labeled *obl* in UD. In the following examples, the English sentence (28) uses *iobj* correctly for the secondary direct object, its Hungarian equivalent (29) uses *iobj* as well, although with respect to the dative form, the *obl* deprel would be correct, as it is in the Czech counterpart (30), which has identical valency for both objects.

(28) "I give the child*[iobj]* a chocolate*[obj]*."

(29) "Csokit*[obj,ACC]* adok a gyereknek*[iobj,DAT]*." (HU)

(30) "Dávám dítěti*[obl:arg,DAT]* čokoládu*[obj,ACC]*."

1.3.4 *obl* - oblique nominal

The oblique deprel, *obl*,[98][99] is the most problematic one regarding the valency. According to the UD guidelines, it covers all other nominal dependents than the ones already mentioned, i.e. the subject and the objects in their typical form.

[96] de Marneffe, Manning, et al. 2021, pp. 273–274
[97] *UD web* 2023, `https://universaldependencies.org/u/dep/iobj.html`
[98] de Marneffe, Manning, et al. 2021, pp. 274–22
[99] *UD web* 2023, `https://universaldependencies.org/u/dep/obl.html`

This means that the dependents, in some grammars traditionally perceived as secondary or indirect objects, should be labeled as oblique in UD, with an *obl* deprel (31), which may be confusing. This relation is of course not used only in combination with the direct object, it is also used in cases where there is no other nominal than the one concerned, which is in a non-typical form for an object, be it the use of a non-typical morphological case (32) or an adposition (33). Another typical example of the use of the *obl* deprel is the agent in the passive voice (36).

(31) "Dala knihu*[ACC,obj]* kamarádce*[DAT,obl:arg]*."
= "She gave the book to her friend."

(32) "Muž podlehl zraněním*[DAT,obl]*."
= "The man succumbed to his injuries."

(33) "I am interested in linguistics*[obl]*."

In all these cases, the oblique relation denotes complements, that would be considered actants, from the FGD point of view. But the oblique relation, as defined in UD, covers also those adjuncts that are expressed by a nominal, whereas adjuncts expressed by an adverb are marked *advmod* (see 1.3.8). The decision to mark complements as *obl* (34) or *advmod* (35) depends obviously on their upostag although the syntactic function is the same (specification of place and time in this case).

(34) "I run to school*[NOUN,obl]* every Monday*[NOUN,obl]*.".

(35) "I run home*[ADV,advmod]* today*[ADV,advmod]*.".

Distinguishing the cases, whether *obl* denotes an actant or an adjunct is a hard task. There is no reliable rule for it, so in this work several heuristics are used (see). However, UD partially allows to specify whether an oblique dependent is an actant or not by deprel extensions: *obl:agent* for the agent in passive constructions (36) and *obl:arg* for other oblique valency arguments (37). But these are not obligatory and they are annotated only in some of the UD treebanks of some languages (e.g. in Czech). But when they are, they can be used for the purpose of valency arguments extraction.

(36) "Jsem sledován policistou*[obl:agent]*."
= "I am being followed by a police officer."

(37) "Zajímám se o jazykovědu*[obl:arg]*."
= "I am interested in linguistics."

1.3.5 *ccomp* - clausal complement

Similarly as nominal subjects, nominal objects can be replaced with subordinate clauses, too. In UD, they are called clausal complements and are marked with a *ccomp* deprel. However, UD does not distinguish a between clauses replacing an *obj*, *iobj* and *obl* dependents. Clauses extending whichever whichever type of

complements take the common *ccomp* deprel. Especially they are used for the reported speech.[100][101] Beside of this, oblique dependents can be represented also by an adverbial clause with an *advcl* deprel (see below). This distinction corresponds actually better to the notions of actants (which are counterparts of clausal complements here) and adjuncts (corresponding rather to the adverbial clauses) from FGD (but here in the form of subordinate clauses), than the analogous relation types for nominal dependents. However, such a different approach to annotation between the nominal and clausal dependents (it can be clearly seen later in the Tab. 1.4) is actually subject to criticism and there are suggestions to change the guidelines so that this problem is removed.[102] Following examples show *ccomp* used as direct object (38), in reported speech (39) and as indirect object (40). For comparison, the last sentence (41) shows a case, where an oblique dependent corresponds to an adverbial clause, being rather an adjunct than an actant.

(38) *ccomp* replacing a direct object in the typical case (ACC) (*obj*)
"Ztratili jsme lampu*[obj]*." = "We lost a lamp."
→ "Ztratili jsme, čeho jsme si vážili*[ccomp]*." = "We lost what we valued."

(39) *ccomp* used in reported speech (*obj*)
"Říká zajímavosti*[obj]*" = "He says interesting things."
→ "Říká, že jde*[ccomp]* domů." = "He says he is going home."

(40) *ccomp* replacing an object in a non-typical case (INS) (*obl*)
"Házím oštěpem*[obl]*." = "I throw a javelin."
→ "Házím, čím mohu*[ccomp]*." = "I throw what I can."

(41) *advcl* replacing an adjunct (*obl*)
"Zpívají na zahradě*[obl]*." = "They sing in the garden."
→ "Zpívají, kamkoli je pozvou*[advcl]*." = "They sing wherever they are invited."

1.3.6 *xcomp* - open clausal complement

Although the clausal complement corresponds to several nominal arguments, it can take not only the *ccomp* deprel, but also *xcomp*, which stands for open clausal complement. It is used in clauses formally lacking their own subject, because it is inherited from the governing clause. The criterion for deciding which deprel to use is whether the clause necessarily must be understood with a subject determined from the higher clause, then the *xcomp* is to be used, or whether there is a possibility to interpret the clause as having a different subject, then *ccomp* is the correct deprel.[103][104] In the examples (1) - (3) above, all clausal complements have

[100] de Marneffe, Manning, et al. 2021, pp. 277–279
[101] *UD web* 2023, `https://universaldependencies.org/u/dep/ccomp.html`
[102] Przepiórkowski and Patejuk 2018
[103] de Marneffe, Manning, et al. 2021, pp. 277–279
[104] *UD web* 2023, `https://universaldependencies.org/u/dep/xcomp.html`

their own subject (although not necessarily expressed on the surface), so they are *ccomp*. The difference between a *ccomp* and *xcomp* is better illustrated in the following sentences that differ in the identification of the subject of the infinitive: in the example (42), the walls will be repaired by someone else than the king, in the sentence (43) it is the sister who is the subject of the writing, i.e.d, both verbs share the same subject. The *xcomp* is also used for secondary predicates (which often corresponds to the notion of verbal attribute in the traditional Czech grammar), but only if the secondary predicate is a core complement of the verb (44) not an adjunct (45). The *xcomp* deprel can correspond to the direct objects (as in the example (43)) or do not have a nominal counterpart at all in the case of the secondary predication (44).

(42) "Král nařídil opravit*[ccomp]* hradby." = "The king ordered to repair*[ccomp]* the walls."

(43) "Moje sestra mu slíbila napsat*[xcomp]* dopis." = "My sister has promised to write*[xcomp]* him a letter."

(44) "Jmenovala ho soudcem*[xcomp]*." = "She named him a judge*[xcomp]*."

(45) "Tančila bosa*[advcl]*." = "She danced barefoot*[advcl]*."

1.3.7 *expl* - expletive

Another UD deprel relevant for the valency is expletive, marked as *expl*, which denotes redundant, formal complements of the predicate that do not semantically satisfy any of its actual argument positions. This includes pronouns used in a position not created by the verb frame (e.g. avalent verbs in languages requiring a subject (46)). Another example are existential expressions in various languages occupying the acting as a specification of place (47), the subject (48) or even both at once (49). The *expl* deprel denotes also reflexive pronouns used in reflexive passive (e.g. in Czech and Slovak, see 2.5.3) (50) or used with verbs requiring them (so called *reflexiva tantum*, only reflexive or inherently reflexive, see also 2.1.8) (51).[105][106] The expletives are by definition not a part of the valency frame of the verb, but sometimes they help to distinguish the meaning of the verb and are therefore relevant for the question of valency. Unlike other nominal complements, expletives do not have a clausal counterpart, as they are actually semantically empty function words.

(46) "It*[expl]* is raining."

(47) "There*[expl]* is a dog*[nsubj]* in the house."

(48) "Es*[expl]* gibt einen Hund*[obj]* in dem Haus." (DE)
= "There is a dog in the house.", lit. "It gives a dog in the house."

(49) "Il*[expl]* y*[expl]* a un chien*[obj]* à la maison." (FR)
= "There is a dog in the house.", lit. "He there has a dog in the house."

(50) "Oběd se*[expl]* vaří." = "The lunch is being cooked."

[105] de Marneffe, Manning, et al. 2021, p. 276

[106] *UD web* 2023, `https://universaldependencies.org/u/dep/expl.html`

(51) “Bojím se*[expl]* o tebe.” = “I am worried about you.”

1.3.8 *advmod* - adverbial modifier

The valency frame (as defined in FGD, 1.2.3) should include not only actants, to which the deprels discussed above mostly correspond, but also obligatory adjuncts. Some of the adjuncts can be expressed by a nominal and thus are marked with an *obl* deprel (examples (52) and (53)), others are in the form of adverbs taking a deprel called adverbial modifier, denoted as *advmod* (examples (54) and (55)).[107] This do not correlate to their obligatoriness, which follows from valency of the verb: they can be obligatory (examples (52) and (54)) or optional (examples (53) and (55)).

(52) “Přišel jsem na zahradu*[obl]*.” = “I came to the garden.”

(53) “Zpíval jsem na zahradě*[obl]*.” = “I sang in the garden.”

(54) “Přišel jsem domů*[advmod]*.” = “I came home.”

(55) “Zpíval jsem doma*[advmod]*.” = “I sang at home.”

1.3.9 *advcl* - adverbial clause

Similarly as most of the deprels discussed above, *advmod* has also a clausal counterpart: *advcl*, meaning adverbial clause modifier.[108][109] Unlike “advmod” it represents all clausal modifiers, so it is also the counterpart of *obl* is some cases. But it still keeps the indistinction in obligatoriness, they can stand in the position of an obligatory (56) or an optional (57) adjunct, depending on the verb.

(56) “Přišli, kamkoli to šlo*[advcl]*.” = “They came wherever it was possible.”

(57) “Zpívali, kdekoli to šlo*[advcl]*.” = “They sang wherever it was possible.”

1.3.10 Nominal and clausal complement correspondence

Tab. 1.4 shows an overview correspondence of deprels of nominal and clausal complements.[110] It does not include expletive which cannot be developed into a clause. Also usage of the *xcomp* deprel for secondary predication do not have a nominal counterpart.

1.3.11 Non-valency deprels

The deprels described above denote complements of predicates (only *advmod* can depend on other modifiers), but not all of them are extracted into the valency frame in this work. Moreover, there are also other deprels depending on predicates, but they are not complements. Among them, there is *aux* as a deprel of

[107] *UD web* 2023, `https://universaldependencies.org/u/dep/advmod.html`
[108] de Marneffe, Manning, et al. 2021, pp. 277–279
[109] *UD web* 2023, `https://universaldependencies.org/u/dep/advcl.html`
[110] de Marneffe, Manning, et al. 2021, p. 258

<table>
<tr><th></th><th>nominal</th><th>clausal</th></tr>
<tr><td></td><td>nsubj</td><td colspan="2">csubj</td></tr>
<tr><td></td><td>obj</td><td rowspan="3">ccomp</td><td>xcomp</td></tr>
<tr><td></td><td>iobj</td></tr>
<tr><td></td><td rowspan="2">obl</td></tr>
<tr><td></td><td rowspan="2">advcl</td></tr>
<tr><td></td><td>advmod</td></tr>
</table>

Table 1.4: Overview of correspondence between nominal and clausal complements. The green marked lines denote actants, the red marked ones adjuncts. The table does not summarize completely all the relations discussed: *expl* deprel does not have a clausal counterpart and vice versa some usages of *xcomp* do not have a nominal one. Moreover the *advmod* deprel is not nominal, strictly speaking, as it is expressed by adverbs. It is put among nominals because it fits into the nominal-clausal dichotomy as a (partial) counterpart of the *advcl* deprel.

auxiliary verbs (with *AUX* upostag),[111][112] *cop* for copula in nominal predicates (where the nominal part is the head of the clause according to the UD principle of semantic priority, see 1.1)[113][114], markers of subordinate clauses *mark* (mostly conjunctions)[115][116] or *compound* deprel denoting parts of multi-word expressions[117] which are also used for various verbal particles (see 2.1.8).

1.3.12 *case* - case marking

When describing a valency frame of a verb, it is necessary to state what is the appropriate form of individual complements. If the form is a morphological feature, it is found among the features of the word, mainly as its *case* feature. But also other words can be part of the form of a complement. Therefore we should look also at the dependents of the verb complements, i.e. two levels under the verb. For nominal complements, the *case* deprel denotes a dependent adposition (preposition or postposition, with the *ADP* upostag).[118][119] It should be not confused with the *case* feature of the complement itself, although both concepts are closely related. Some languages use morphological cases, other use adpositions, but both ways are equivalent and corresponding, so they share a common name in UD. Following examples shows the analogy between the *case* feature (58) and the *case* deprel (59).

(58) "Majitel firmy*[features:case=GEN]* krájí chléb nožem*[features:case=INS]*."

[111] de Marneffe, Manning, et al. 2021, pp. 19–20
[112] *UD web* 2023, `https://universaldependencies.org/u/dep/aux.html`
[113] de Marneffe, Manning, et al. 2021, pp. 273–274
[114] *UD web* 2023, `https://universaldependencies.org/u/dep/cop.html`
[115] de Marneffe, Manning, et al. 2021, p. 278
[116] *UD web* 2023, `https://universaldependencies.org/u/dep/mark.html`
[117] *UD web* 2023, `https://universaldependencies.org/u/dep/compound.html`
[118] de Marneffe, Manning, et al. 2021, pp. 269–270
[119] *UD web* 2023, `https://universaldependencies.org/u/dep/case.html`

(59) = "The owner of*[deprel=case]* the company cuts the bread with*[deprel=case]* a knife."

1.3.13 *mark* - marker

Another dependent on a verb complement we must take into account is the already mentioned marker of a subordinate clause (*mark*). It is used with clausal complements and denotes conjunctions joining the complement to its governing predicate.[120] Note that for the task of valency frame extraction, the *mark* dependents of the clausal complements play a role, whereas the *mark* dependents on the head verb of the frame itself do not, as they describe the relation of the verbs to higher sentence levels, which is not related to its valency frame. In the sentence (60), the middle verb *say* depends on *instructed* and governs *work*. The first marker *to* connects the verb *say* to its head *instructed*, so it is not important for the valency frame of the *say*. On the contrary, the marker *that* connecting the verb *work* to its head *say* plays an important role in the valency of *say* - it says in which form (by which conjunction) should be its complements connected to it.

(60) "We instructed him to*[mark]* say that*[mark]* it does not work."

1.3.14 *conj* - conjunct and coordination in UD

Coordination is a linguistic phenomenon, when multiple referents are grouped and they occupy the same position in the sentence together. They are typically separated by a coordinating conjunction (e.g. *and*, *or* in English) or a punctuation mark (a comma). The members of the coordination are called conjuncts. Various dependency linguistic theories differ in the representation of coordination; there are two main approaches. The coordinating conjunction or punctuation can serve as a technical head node of the whole structure and all conjuncts depend on it. Alternatively, one of the conjuncts (typically the first or the last one) may serve as the head with the rest of the conjuncts attached below it. The former approach is used e.g. in PDT, but UD uses the latter one (with the first conjunct being the head). It complies better with the UD principle of semantic primacy and preserves the true relation of the conjunction or punctuation. Its disadvantage lies is different treatment of the heading conjunct and the other conjuncts.[121]

The heading conjunct is labeled with the proper deprel according to the role of the whole coordination structure in the sentence. The other conjuncts depend on it with a *conj* deprel.[122] During the valency frame extraction, the coordination does not cause any trouble in the case of coordinated arguments, because for extraction of the argument including its description the heading conjunct should suffice; for the verb, there is no difference between having one nominal or multiple coordinated nominals as arguments. A problem occurs, when there are coordinated verbs with some of their arguments shared, which is not marked in the

[120] *UD web* 2023, `https://universaldependencies.org/u/dep/mark.html`
[121] *UD web* 2023, `https://universaldependencies.org/v2/coordination.html`
[122] *UD web* 2023, `https://universaldependencies.org/u/dep/conj.html`

annotation and the shard arguments depend only on the heading conjunct verb. In that case, the extraction of valency arguments is difficult. Possible solutions of this issue is described in 2.4.3.

1.4 Data and tools

1.4.1 Python and Shell

The program part of the work consist of multiple scripts written mostly in Python. Python is a modern, comfortable high-level language, that is appropriate for multiple various computer tasks. It gains still more and more attention because of the large amount of helpful libraries, especially related to machine learning and data analysis. It is extensively used in computational linguistics, too. Udapi, interface for accessing UD data, used also in this work (see 1.4.8) works in Python, for these reasons I decided to develop the program in Python, too.

However, the scripts run directly by the user are written in Shell. The main motivation for this was that several tools described below are meant to be run from the command line, rather than from a Python script.

1.4.2 CoNLL-U format

UD defines the attributes of the tokens, their possible values and principles for their assignment, but not the particular format in which this information should be stored. However, there is a widely used format, in which UD treebanks are stored, called CoNLL-U.[123] It is a revised version of an older CoNLL-X[124] format, adapted to the needs of UD. The shortcut stands for "Conference on Computational Natural Language Learning", the "X" in the original format stood for the tenth such conference which was dealing with a task on multilingual dependency parsing, the "U" in the actual format stands for "universal". A file in CoNLL-U format (with the *.conllu* suffix) contains three types of lines: blank lines used to separate sentences, comment lines (starting with the # character) and record lines. Each sentence is preceded by at least one comment line with a sentence identifier. The sentence representation consists of a sequence of record lines, each for one token (there are exceptions concerning multi-word tokens or artificial nodes). A token record is divided into ten columns separated by a tab:

1. ID – identifier of the token, i.e. its order in the sentence (numbered from 1)
2. FORM – the token in the form in which it appears in the sentence
3. LEMMA – the token in its basic form (it might differ from the actual form for inflected words)
4. UPOSTAG – *upostag*, universal part-of-speech tag, defined in UD (see 1.1)
5. XPOSTAG – a language-specific part-of-speech tag (e.g. from the original corpus), not used in this work

[123] *UD web* 2023, `https://universaldependencies.org/format.html`
[124] Buchholz and Marsi 2006

6. FEATS – list of features (lexical or morphological, see 1.1) of the given token in the attribute-value form, defined in UD, but the value might have also a language-specific part

7. HEAD – identifier of the head token (on which this token is dependent), 0 if the token is a tree root

8. DEPREL – *deprel*, type of dependency relation to the head, defined in UD (see 1.1)

9. DEPS – list of secondary dependencies and other relations on other sentence members in the head:deprel form, not used in this work

10. MISC – any other annotation, not used in this work

An example of CoNLL-U format is shown on the Tab. 1.5. It is the same sentence as shown in Fig. 1.1 in 1.1 as an example of dependency tree.

1.4.3 UD treebanks, PUD

As of 2023, the UD project includes over 200 annotated treebanks for over 100 languages. The treebanks differ considerably in size (number of sentences or tokens), text source domain (legal or academic texts, Wikipedia articles, news, blog texts etc.) and the extent of annotation (lemmas, upostags, features and relations – see CoNLL-U 1.4.2). Many of the treebanks were originally created with a non-UD (mostly manual) annotation and then converted automatically to UD. There are also several treebanks created for the UD project and thus annotated originally in the UD style.

Among them, the Parallel Universal Dependencies treebanks (PUD) are especially important. They are a set of parallel treebanks, each consisting of 1000 sentences. The PUD Treebank exists for many languages including English[125] and Czech[126], used in this work. Most of the sentences comes originally from English, the annotation is made natively in UD.

1.4.4 JRC-Acquis

Although the UD project includes many treebanks, most of them are not parallel. In order to obtain a lot of parallel text for many language pairs it is necessary to reach for other, non-UD parallel corpora and annotate them in UD style via UDPipe (1.4.6), which causes lower annotation quality though, than the manual annotation. For this purpose, I decided to used JRC-Acquis corpora in this work.[127][128]

The Joint Research Centre (JRC) of the European Commission releases a large collection of selected legislature applicable in the European Union (EU), called Acquis. The purpose of the JRC-Acquis is to allow cross-lingual research

[125] *UD web* 2023, https://universaldependencies.org/treebanks/en_pud/index.html

[126] *UD web* 2023, https://universaldependencies.org/treebanks/cs_pud/index.html

[127] Steinberger et al. 2006, corpus description `https://joint-research-centre.ec.europa.eu/language-technology-resources/jrc-acquis_en`

[128] Steinberger et al. 2006, corpus download `https://wt-public.emm4u.eu/Acquis/JRC-Acquis.3.0/alignmentsHunAlign/index.html`

# text = My sister gave me a very nice book, when I asked her.									
1	My	my	DET	_	Number=Sing| Person=1| Poss=Yes| PronType=Prs	2	nmod:poss	_	_
2	sister	sister	NOUN	_	Number=Sing	3	nsubj	_	_
3	gave	give	VERB	_	Mood=Ind| Tense=Past| VerbForm=Fin	0	root	_	_
4	me	I	PRON	_	Case=Acc| Number=Sing| Person=1| PronType=Prs	3	iobj	_	_
5	a	a	DET	_	Definite=Ind| PronType=Art	8	det	_	_
6	very	very	ADV	_	_	7	advmod	_	_
7	nice	nice	ADJ	_	Degree=Pos	8	amod	_	_
8	book	book	NOUN	_	Number=Sing	3	obj	_	_
9	,	,	PUNCT	_	_	12	punct	_	_
10	when	when	SCONJ	_	Case=Nom	12	mark	_	_
11	I	I	PRON	_	Case=Nom| Number=Sing| Person=1| PronType=Prs	12	nsubj	_	_
12	asked	ask	VERB	_	Mood=Ind| Tense=Past| VerbForm=Fin	3	advcl	_	_
13	her	she	PRON	_	Case=Acc| Gender=Fem| Number=Sing| Person=3| PronType=Prs	12	obj	_	_
14	.	.	PUNCT	_	_	3	punct	_	_

Table 1.5: Example of a sentence annotated in CoNLL-U format. Empty fields contain an underscore.

on many languages. The collection includes texts in 22 of the 24 official languages of European union (currently the Irish and Croatian are missing). The texts are taken from the EU documents from 1950s until now, but not all documents are available in all languages. On average, there are ca. 23,000 documents for each language, with approximately 30,000,000 words. The corpus is equipped with a sentence alignment for each language pair. There are several versions of the alignments available, depending on the tool used. I used the alignments created by HunAlign tool. Number of sentence pairs differ for each language pair, but all language pairs used in this work have nearly 1,000,000 sentence pairs. There is a corpus file for each language and an alignment file for each language pair. A tool creating a parallel corpus based on the given monolingual corpora with their alignment file is also provided. All the files are in *.xml* format, which requires further processing to obtain a plain text necessary for the UDPipe.

Acquis corpora offer a large amount of multilingual parallel data, which allows to perform a lot of cross-lingual research on them and makes them appropriate also for this work. In fact, it is the biggest existing parallel corpus, if we consider both its size and number of languages. Another major advantage is the possibility to obtain any language pair for the languages contained, including also rare language pairs, for which there are no specifically built bilingual parallel corpora (e.g. Maltese-Estonian corpus).

On the other hand, they have also several problematic aspects. The necessity of further automatic UD annotation for the purpose of this work was already mentioned. The texts are predominantly of legal character (see the examples (61) and (62)) in comparison with the texts contained in UD corpora (1.4.3) which are often taken from different sources and make the whole collection more representative. Another problem is that the sentence alignment in Acquis is done automatically, which sometimes leads to incorrect sentence pairs. Typically one sentence on one side correspond to two sentences on the other side and several following sentences are then shifted, until a reverse case of two sentences corresponding to one restores the correct alignment.

Lastly, The parallel texts for individual languages are translations from one original language, usually English. However, the translations are often very loose, which leads to usage of different constructions in the parallel sentences and makes building a valency dictionary more difficult. A frequent case of different translations is nominalization: a meaning expressed in one language with a verb is in the other language expressed by a nominal phrase. In the Czech-English phrase pair (61), the Czech phrase avoids use of a verb and makes linking of the verb *to cover* on the English side impossible. In the Czech-Slovak phrase pair (62), there is an extra verb *ustanovovať = to establish* in the Slovak version, that has no counterpart on the Czech side.

(61) "v oblasti působnosti této směrnice", (CS)
= "in the field of activity of this directive"
∼ "in the field covered by this directive" (EN)

(62) "podle této dohody" (CS)
= "according to this agreement"
∼ "ktoré ustanovuje táto dohoda" (SK)
= "which this agreement establishes"

1.4.5 Vallex and related valency dictionaries

For the purpose of evaluation of the extracted valency dictionaries, particularly for Czech and English, existing valency dictionaries are used. Monolingual Czech dictionary is evaluated using valency dictionary called Vallex. For English, an analogous dictionary named EngVallex is used. Most importantly, there is also a bilingual valency dictionary called CzEngVallex with alignment of frames and frame arguments. In this dictionary, the corresponding valency frames on both sides are linked together. It inspired the form, in which are extracted bilingual valency dictionaries represented in this work. All the three dictionaries share the same theoretical grounds, a similar structure, annotation, *xml* format and also name. They will be sometimes together referred in this work as Vallex dictionaries.

Vallex dictionaries are based on the valency theory of FGD. They are being developed and maintained since 2008 at the Faculty of Mathematics and Physics of Charles University. It adopts its notion of valency frame as a set of all actants and obligatory adjuncts. A central structure in Vallex is a lexeme, which typically represents two verbs differing in their aspect: perfective/imperfective (*odpovědět*/*odpovídat*) or imperfective/iterative (*čistit*/*čistívat*). Each lexeme consists of several lexical units which may differ in meaning and in valency frame. The valency frame of every lexical unit is a sequence of valency complements described by their functor, their form (case, preposition + case) and their obligatoriness (*obl* – obligatory, *opt* – optional). Vallex include also typical optional adjuncts (typ – typical), which are not part of the valency frame (e.g. for the lexeme *kupovat* = *to buy*, there is a typical optional adjunct *RCMP.za+4* describing the price in the accusative with the preposition *za*).

There are multiple versions of Vallex, the newest being Vallex 4.5 (2022).[129] In this work, an older version, Vallex 3.0 published in 2016, is used.[130] In the case of EngVallex, this work uses the first version from year 2014,[131] although there is a newer one EngVallex 2.0 published in 2021.[132] The bilingual valency dictionary CzEngVallex dates back to the year 2015.[133]

They are used at several places in this work. Firstly, the extracted monolingual Czech and English frames are matched to their counterparts in Vallex and EngVallex and the statistics regarding this matching are presented and discussed (2.7.13). Secondly, the extracted frames of the bilingual Czech-English valency dictionary are matched to the corresponding frames in Czech and English parts of CzEngVallex and the success of the bilingual linking of the extracted dictionary is measured according the the frame pairs in CzEngVallex (**??**). Lastly, the Vallex is used in the exploring of possibilities of valency dictionary projection, and the information present in Vallex, that cannot be extracted from a UD annotated treebank, is projected to an extracted valency dictionary for another language via the linking of the extracted frames in the bilingual valency dictionary (**??**).

[129]Lopatková, Kettnerová, Mírovský, et al. 2022, `https://ufal.mff.cuni.cz/vallex/4.5/`

[130]Lopatková, Kettnerová, Bejček, et al. 2016, `https://ufal.mff.cuni.cz/vallex/3.0/`

[131]Cinková et al. 2014, `https://lindat.mff.cuni.cz/services/EngVallex/`

[132]Cinková et al. 2021, `https://lindat.mff.cuni.cz/services/EngVallex20/`

[133]Urešová, Fučíková, and Šindlerová 2016, `https://lindat.mff.cuni.cz/services/CzEngVallex/`

1.4.6 UDPipe

UDPipe is a tool for automatic morphological and syntactic sentence analysis and annotation according to UD principles.[134] It can perform tokenization, morphological tagging and syntactic parsing and outputs annotated text in CoNNL-U format. It was created by combining several separate tools (Parsito, MorphoDiTa etc.). It works mainly on the principle of deep machine learning and contains models for different languages trained on their UD corpora. There are several versions of UDPipe, I work with the latest models of UDPipe 1.[135]

In this work, UDPipe is used two times during the preparation for the actual valency frames extraction. The first time, it used only for tokenization of sentences obtained from parallel corpora. The tokenized text is then passed to Fast aligner (see 1.4.7) for word-alignment. The tokenized text is then morphologically and syntactically analyzed via the second use of UDPipe.

As an automatic tool, UDPipe makes mistakes in the annotation. Incorrect assignment of lemmas, upostags, deprels and heads are relevant for this work, because it can worsten the quality of the frame linking methods which use this information.

Incorrect lemmas are either non-existing forms (in English *referr* instead of *refer*, *conven* instead of convene, *apprive* instead of *approve*, in Slovak *prijímajúť* instead of *prijímať*, meaning *to accept*) or forms of a different lexeme (a Czech form *znějí*, *they sound* got a wrong lemma *znát*, *to know* instead of *znít*; the corrsesponding finite form from the verb *znát* would be *znají*). The cases of incorrect upostags include marking verbs with other parts of speech because of wrong tagging (*comply* as an adverb) or disambiguation (word *invite* and *means* incorrectly marked as nouns in the place concerned) or vice versa marking other words as verbs (Slovak *kategória* meaning the noun *category* or often the letters *a*, *b* etc. enumerating items in a list).

1.4.7 fast_align

fast_align is a tool for word alignment of tokens in parallel sentences.[136] It is unsupervised and works on the basis of common occurrences of words in the corresponding sentences. The larger the parallel corpus is, the better results should be produced.

The input sentences must be tokenized (tokens including punctuation must be separated by spaces) and the corresponding sentences must be written on the same line (separated by three vertical bars). The output is a sequence of pairs of tokens represented by their order in the sentence (numbered from zero) for each pair of sentences. The algorithm always maps tokens from one sentence onto the tokens of the other. It may leave some tokens of the second sentence unpaired (naturally, as the number of its tokens can be higher than in the first sentence) or, on the other hand, map many tokens from the first sentence onto one token from the other (not only in cases when the its token number is lower). To obtain a balanced word alignment, the process must be run twice, once in each direction. It may even leave several tokens from the first sentence unpaired, so it is not a

[134]Straka and Straková 2017
[135]Straka and Straková 2019
[136]Dyer, Chahuneau, and Smith 2013

complete mapping, in fact. A token can be mapped to no or to one other token and any number of other tokens (incl. zero) can be mapped to it. The tool allows to combine the outputs by union or the intersection, but more complex heuristics are used in this work (see **??**)

1.4.8 Udapi

Udapi is an interface for easy work with data annotated according to UD.[137] There are several versions of the interface according to the programming language (beside a Python version used in this work, there are also versions for Perl and Java). When processing an input CoNLL-U file, Udapi creates a data structure that allows further processing on several levels: on the level of document (the whole file), bundle (a group of sentences associated together, e.g. corresponding sentences in a parallel corpus), root (a tree of a sentence depicting its actual dependency structure) and node (a token having its record line in the CoNLL-U file). The nodes allow an easy access to all necessary data, such us the grammatical properties of the expression, its head, its dependent tokens or its order in the sentence. The interface allows also changes in the structure.

The preferred way to use Udapi is to build scenarios consisting of a sequence of several processing blocks. Blocks then run processing on the mentioned processing levels gradually from above and the user of the interface is invited to overload the default methods on the level, where his processing should take place. There is already a prepared reading block for loading the structure from a CoNLL-U file. The usage of Udapi in this work lies in a scenario consisting of this reading block and of the newly created valency block that extracts valency frames for the verbs. No writing block is necessary as the structure is not changed by the valency frames extraction nor actually needed anymore. The extracted valency frames are stored in another structure (see C) and saved into a binary file.

1.5 Related works

Up to now, there is no tool designed specifically for automatic creation of multilingual valency dictionary based on UD. However, particular parts of this aim were already discussed. There are projects trying to modify UD internally so that it contained more semantic information including valency and there are also efforts to create multilingual valency dictionaries outside the UD project. The bilingual Czech-English valency dictionary CzEngVallex was already mentioned (see 1.4.5) and its is used in this work for evaluation.

1.5.1 Enhanced Universal Dependencies

Regarding the internal discussion in UD project, the first step was the introduction of *enhanced* dependencies.[138][139] This extension of UD allows to add artificial nodes for elided predicates (but not for elided subjects and objects, such as

[137] Popel, Žabokrtský, and Vojtek 2017

[138] Schuster and Manning 2016

[139] *UD web* 2023, `https://universaldependencies.org/u/overview/enhanced-syntax.html`

pro-drop pronouns) or additional secondary relations to capture other relations between sentence members, so that it would not violate the basic tree structure. They connect conjuncts in a coordination group (other that the first one) directly with the head of the whole group or subjects of a control or phrasal verbs with an infinitive dependent on them as also their infinitive (e.g. "in Susan wants to run", the secondary relation would attach *Susan* as a subject of *run*). The enhanced dependencies also encourage to use specifications of deprels (written after a colon), among other things also for specification of a case of an argument (e.g. *obl:dat* instead of a bare *obl*).

1.5.2 Deep Universal Dependencies

A further step in this regard was later made by presenting Deep Universal Dependencies, introduce deeper semantic annotation into UD.[140] It focuses on capturing valency and contains assignment of verb arguments to the head verb including argument numbering according to their salience (similarly as Tesnière proposed). The verb is marked with a verb lemma (possibly enriched with some additional information) as a frame identifier or even with a reference to a valency dictionary, if this is available. The arguments are not labeled with a semantic role, but this can be done also in the valency dictionary. Deep UD uses an extension of CoNNL-U format, which uses two new columns and for the verb nodes presents their frame identifier and list of arguments (in an *argument_number:node_id* form). This approach enables to attach one node to several verbs as their argument, for example in a case of verb coordination.[141]

Outside the UD, there are several projects trying to build multilingual resources containing various semantic information including valency. Most of the works trying to automatically extract valency information were focused on English, but there was also an attempt for Czech working with PDT, long before that UD and even before Vallex were created. The authors tested three statistical methods for valency frames extraction with subsequent frame reduction.[142]

1.5.3 SynSemClass

An important contemporaneous project related to the presented work is SynSemClass,[143] being developed at Institute of Formal and Applied Linguistics on the Faculty of Mathematics and Physics at the Charles University in Prague. It is a multilingual semantic ontology assigning verbs, or rather individual verb senses represented by different valency frames,[144] to synonym classes, which are sets of verbs with the same or very similar meaning. They define semantic roles which should be fulfilled by the class members with their arguments regardless of their actual syntactic realization.[145] The idea is that it should be possible to put each verb in each language into a synonym class, even map its arguments to the seman-

[140] Droganova and Zeman 2019
[141] Droganova and Zeman 2019, pp. 148–149
[142] Sarkar and Zeman 2000
[143] Urešová, Zaczynska, et al. 2022, `https://ufal.mff.cuni.cz/synsemclass`
[144] Urešová, Fučíková, Hajičová, et al. 2019, p. 41
[145] Urešová, Fučíková, Hajičová, et al. 2019, p. 38

tic roles of the class.[146] Originally, it began as a bilingual net between Czech and English, later German was added and extensions to more languages are to come. The classes are mapped to many existing resources including some of the Vallex dictionaries.[147] In the future, more languages are expected to be included[148], what will require necessary modifications of the lexicon like adding, merging or splitting synonym classes or changing their semantic roles.[149][150]. The annotation will also include more information, like semantic roles labels or hierarchy of classes and roles. Extension to nouns and adjectives is also planned.[151]

The multilingual ambitions of the project were inspired by the success of UD,[152]. Examining the influence of syntactic realization of the core arguments in UD on the assigning a verb to a synonym class was a part of the project, too.[153]

The multilinguality and mapping of verbs and their arguments among different languages are aims that the SynSemClass project has in common with the presented work. Unlike SynSemClass, this work operates only with the syntactic information contained in UD treebanks and does not use the semantic information from other sources. Also the concept of synonym classes is more general than individual links between two valency frames in an extracted valency dictionary (the synonym class covers synonymy also within one language). The presented valency frames extraction works automatically and aims on being run repeatedly the resources will improve, while building the SynSemClass lexicon is a gradual process, partially automatic, but including also substantial manual corrections.[154]

1.5.4 Universal Proposition Bank

Another important related project is Universal Proposition Bank.[155] It is a multilingual collection of propbanks with projected semantic labels. As there are manually annotated propbanks for only few languages, because the annotation is expensive, the solution this project offers is to project it from an existing proposition bank (particularly the English PropBank) via parallel corpora. Given a parallel corpus of English and a target language, the English part is annotated with semantic roles of predicates using a labeler trained on the English PropBank. The semantic roles are then projected through the word alignment into the target language. Beside of this automatic projection, the project includes also manual semantic annotation for several languages. The actual version of Universal Proposition Bank 2.0 contains high-quality semantically labelled propbanks for 23 languages from 8 languages families and it is planned to grow.[156]

The idea of projecting semantic labels is used on this work as well, in the chapter **??**. While Universal Poposition Bank uses automatic semantic labelers to annotate the the source language part of the parallel treeebank, the presented

[146]Urešová, Zaczynska, et al. 2022, p. 1338
[147]Urešová, Fučíková, Hajičová, et al. 2019, pp. 38–39
[148]Urešová, Fučíková, Hajičová, et al. 2019, p. 38
[149]Urešová, Fučíková, Hajičová, et al. 2019, p. 41
[150]Urešová, Zaczynska, et al. 2022, p. 1338
[151]Urešová, Zaczynska, et al. 2022, p. 1339
[152]Urešová, Zaczynska, et al. 2022, p. 1338
[153]Urešová, Fučíková, Hajič, et al. 2018, p. 75
[154]Urešová, Fučíková, Hajičová, et al. 2019, p. 38
[155]Jindal et al. 2022, `https://universalpropositions.github.io/`
[156]Jindal et al. 2022, pp. 1700–1701, 1707

programs projects works solely with the dictionaries: it transfers the semantic role labels through via extracted valency frames of the source language (particularly Czech) to the frames of the target language, that are linked with them.

1.5.5 Other related works

The extraction of valency dictionaries is not a new idea. A lot of work was done in this area even before UD started to exist. Many of such efforts worked with Prague Dependency Treebank.[157][158][159]. This work aims to follow up and try something similar, but for Universal Dependencies.

I would like to mention two more projects. The first is a Latvian treebank with multiple layers of automatically obtained annotation. One of the layers is the UD annotation and another contains semantic information including semantic roles as in FrameNet.[160] The second project is a successful attempt to develop a valency parser based on UD treebanks.[161].

[157]Bojar 2002
[158]Žabokrtský 2005
[159]Sarkar and Zeman 2000
[160]Gruzitis, Nespore-Berzkalne, and Saulite 2018
[161]Shi and Lee 2018

Chapter 2

Monolingual valency frames extraction

This chapter deals with extraction of valency dictionary from a treebank for one language. After several introductory paragraphs explaining the nature of this task (2.0.1 - 2.0.7), the basic solution described in more detail (2.1 - 2.2) and extensions treating more complex phenomena (2.3 - 2.6). At the end, the evaluation of whole task by various means is presented (2.7 - 2.7.13).

2.0.1 Monolingual extraction vs. cross-lingual linking

The main aim of the program is to be able to create a multilingual valency dictionary based on given corpora in UD format. Creating such a multilingual dictionary requires the creation of monolingual dictionaries for the individual languages, too, although in the case of the extraction from a parallel corpus, it is possible that the extracted frame in one language helps to extract the frame from the parallel sentence in the other language. Still, this work is divided into the two main parts: extraction of valency frames for only one language, which is described in this chapter, and linking extracted valency frames across multiple languages, which is the topic of the next chapter 3. Each part describes main methods considered for the given task and evaluates them.

2.0.2 Rule based approach

All algorithms described in this work for both monolingual valency frames extraction and their cross-lingual linking are solely rule-based, without any use of statistical or deep machine learning methods. The main advantage of the rule-based approach is that it can be used even for small treebanks, whereas the machine learning needs large collections of training data. If sufficient data are provided, the machine learning methods achieve noticeably better results. For the task of creating multilingual valency dictionary, such data are available only for a few languages, so the rule-based approach enables more universal use of the program. On the other hand, machine learning methods are quite language-independent in general, while linguistic rules need not to - some of them may hold for all languages, but others are language-specific (both types of rules are specified separately in different modules - see below). Another motivation for the

use of the rules is that they have strong linguistic foundation and the connection between a particular rule and the results may be a valuable linguistic contribution. Rule-based are only proper algorithms, presented in this work, but many tools and data the program uses externally are based on machine learning (see 1.4). Neverthless, some practices from machine learning are used in 3.2 for fitting the parameters and evaluation of the frame-linking system.

2.0.3 Use of the UD annotation

In order to extract the valency frames from data, we need them annotated, so that we know which words are verbs, what their dependents are and what the relation of the dependents to the verb is. Because the program aims at language independence, the use of UD as the required annotation style is an appropriate choice. To obtain a UD annotated corpus we must either use one of the available UD treebanks (see 1.4.3) or annotate a text corpus using an automatic annotation tool like UDPipe (see 1.4.6). Yet another option is to use a treebank annotated in a non-UD style and make an automatic conversion. This is not used in this work, because it is not easy to create a good conversion program and because such a program would work only for a particular treebank a could not be used universally. In fact, many UD treebanks were created this way, by automatic or semi-automatic conversion from an existing treebank annotated in another style.

2.0.4 Difference from FGD frames

In the previous chapter several valency theories were discussed (see 1.2). Although the notion of the valency frame used in this work is inspired by the FGD concept of valency, it is necessary to come up with our own scheme, because the UD works with different annotation that FGD uses and there may not be all the necessary information for the extraction of the proper FGD valency frames. The differences concern mostly the selection of the arguments. The actual form of the valency frames used in this work is discussed in 2.2 in more detail.

2.0.5 Basic algorithm

The monolingual valency frames extraction is implemented in several classes, called *extractors*. All of them are (directly or indirectly) inherited from Udapi block (see 1.4.8) and are thus part of the Udapi processing scenario as its second and last part after a reading block. The basic extractor implements the technical part of the extraction. When it needs to take a linguistic decision, it chooses the simplest one. It is used as baseline for measuring the contribution of other, linguistically more advanced extractors.

A valency frame consists of two main parts: the verb and its arguments. Accordingly, the algorithm for valency frames extraction of the basic extractor is quite simple and consists of the two main parts: selecting verbs, for which the frames will be extracted (described in 2.1), and choosing their valency arguments (described in 2.2). The algorithm sequentially processes sentences of the given corpus and each time it finds an occurrence of a verb, it extracts its arguments creating a valency frame. These parts do not need to be necessarily independent: it is imaginable that the decision on whether a word is a verb would depend on

processing its dependents, but still they are two clearly separate tasks. After the frame is created, it is added to the valency dictionary (shortly described below in 2.0.7). The overview of this basic algorithm can be seen in Algorithm 1.

Algorithm 1 Basic frame extraction algorithm

Input: *treebank*
Output: *dictionary*

```
1: dictionary ← empty dictionary
2: for sentence in treebank do
3:     for node in sentence.nodes do
4:         if node.upostag = "VERB" then
5:             frame ← create_frame(node)
6:             for child_node in node.children do
7:                 if child_node is appropriate argument then
8:                     argument ← create_argument(child_node)
9:                     frame.add_argument(argument)
10:            dictionary.add_frame(frame)
```

2.0.6 Extending extractors

Although this works aims at universality and language independence we can suppose that an approach including language-specific techniques could reach better results. This is mostly because of differences between various languages themselves, but often also due to different annotation customs for individual languages. Despite that UD aims at creating a unified standard for description of all languages, this is not observed absolutely and the traditional annotation is often kept also here. Examples of such differences between Czech and English are the annotation of modal verbs (see 2.5.2 and 2.6.2) or participles (see 2.5.1 and 2.6.4). So the program counts on the possibility to add language-specific extension to the basic extractor. This solution enables to improve the specific extractors without changing the general algorithm, to try different extensions for the same language or to add extractors for new languages. For languages without a specific extractor, the general solution still can be used, but it is desirable to have one for each language, on which the valency frames extraction is applied. In this work, I present three language-specific extractors: for Czech, Slovak and English. Moreover, not all extending extractors are language-specific, I present also a language-independent extractor that extends the basic algorithm. All extending extractors are inherited from the basic one and are described in 2.3 in detail.

2.0.7 Adding a frame into the dictionary

After a new valency frame is extracted, the last step of the algorithm is adding it into the valency dictionary being created. The new valency frame is created as two objects: *frame instance* representing the particular occurrence of the frame in the data and *frame type* which represents the abstraction of the frame that can be realized in multiple frame instances (see C).) The algorithm checks, whether

the frame type has been already found before and is contained among the frame types of the respective verb in the dictionary. If no, the new frame type is added into the dictionary with its instance. If yes, the algorithm only adds the frame instance among the instances of the corresponding frame type in the dictionary and the new frame type is deleted. For this purpose, the comparing of two frame types is implemented, which decides whether two frame types are identical or not.

Two valency frames are identical, if they have identical verb lemma (they are frames of the same verb) and identical arguments, i.e. they have the same number of arguments and there is a perfect matching between the two sets of arguments, where the arguments of one pair are identical. Two frame arguments are identical, if they agree in all defining properties (described in 2.2 in more detail): deprel, *Case* and *VerbForm* features and lemmas of their *case* and *mark* dependants.

2.1 Selection of verbs

During the sentence processing, individual words (nodes of the sentence tree structure) are examined, whether they can be considered a verb forming a valency frame. Having UD annotated data, the straightforward approach, which is in most situations kept, is to take nodes with *VERB* upostag. However, two main problematic areas show up. Sometimes it is not clear what is and is not a verb, which is further analyzed in 2.1.1. Moreover many words, formally being verbs, have actually a rather auxiliary function and may not be appropriate to appear in the valency dictionary. This issue is discussed in 2.1.2 - 2.1.7 in more detail. Besides these two main problems, there are also other issues, like verbal particles - words that usually come with verbs (see 2.1.8), and there may be of course also mistakes in the treebank annotation that complicate the straightforward verb selection.

2.1.1 Verbs vs. other parts of speech

As it was already mentioned (1.2.3) the phenomenon of valency relates not only to verbs, but also to nouns, adjectives and adverbs. However, the extent of valency among the non-verbal parts of of speech is limited and they have their own valency frames mostly in cases when they are derived from verbs. Moreover, the research on the verbal valency has a long tradition, whereas the valency of the other parts of speech still deserves further examining. In this work, I focus only on the verbal valency.

First we need to know, which words in the text are verbs. Having a UD-annotated treebank, this task seems straightforward, as there are upostags marked for each word and one of them is *VERB*. But sometimes it is questionable, whether a word is a verb, because of its form or meaning. In many languages, there are word forms on the border of verbs and other parts of speech (namely nouns, adjectives or adverbs), such as infinitives, participles, gerunds, gerundives, transgressives, verbal nouns and adjectives etc. They are annotated either as verbs or as the corresponding non-verbal category, depending on linguistic tradition of the language or a decision of treebank annotation authors, but they probably maintain the valency of the verb, so taking or not taking them into account should

not make any problems (if we have a treebank big enough so that their exclusion will not cause any lack of information).

Czech verbal nouns (e.g. *zpívání = singing*), although derived from verbs, they behave completely as nouns and are annotated accordingly in UD (1). Infinitives (e.g. *zpívat = to sing*) can in many cases hold nominal positions in a sentence, such as object, and still keep the *VERB* upostag (2). Other verb forms can behave as adverbs and be labeled *advcl*, like Czech transgressives (3).

(1) "Požadujeme zpívání$_{[NOUN,obj]}$ moderních písní."
= "We require singing$_{[NOUN,obj]}$ of modern songs."

(2) "Požadujeme zpívat$_{[VERB,xcomp]}$ moderní písně."
= "We require to sing$_{[VERB,xcomp]}$ modern songs."

(3) "Šel jsem domů zpívaje$_{[VERB,advcl]}$."
= "I went home singing$_{[VERB,advcl]}$."

A problem occurs if such forms are annotated with various lemmas, depending on their upostag. This would lead to several distinct records in the valency dictionary with the same arguments, but differing in lemmas. This is the case of Czech, where verbal adjectives preserve the verbal valency, but are marked with *ADJ* upostag and corresponding adjectival lemma, e.g. *ukrytého*, *of the covered one* with lemma *ukrytý*, *covered* instead of the infinitive *ukrýt to cover* (see also 2.5.1 for more details). English counterparts of these cases are annotated as verbs, which complicates the cross-lingual nature of the solution. In manually annotated Czech UD treebanks, the information about the original verbs is contained in the last column of the annotation, but this information is not used to train the UDPipe models, which means the verbal lemmas would not be automatically annotated on new texts. Moreover, this additional annotation is specific for Czech and using it would not solve the problem generally. The solution used in this work is to include them, if possible, even if it leads to duplicate records in the dictionary. The aim is not to lose the data contained in the corpus.

2.1.2 Autosemantic vs. synsemantic verbs

The described approach has a main problem with words, that are formally verbs, but they do not have a full meaning on their own. In opposition to autosemantic words, which having their own meaning, these words are called synsemantic and they mostly help other words to obtain the meaning desired. However, the border between the two groups is fuzzy. There are several groups of verbs which differ in their position on the synsemantic-autosemantic axis and it is not clear if we should include the particular group of them into the valency dictionary. From the perspective of UD, these groups are annotated differently: the ones rather autosemantic bear the *VERB* upostag, if they are considered more synsemantic, they are marked *AUX*. However, the annotation differs across languages and even in one language, the same verb can be labelled in different situations in both ways.

I will distinguish and discuss following categories of such verbs (ordered from the more synsemantic ones to the more autosemantic): auxiliary verbs 2.1.3 in the narrow sense (*you **do** not sing*), copulae 2.1.4 (*he **is** good*), modal verbs 2.1.5 (*she **can** swim*), phase verbs 2.1.6 (*I **stopped** smoking*) and light verbs 2.1.7 (*she*

took *a shower*), which are already very close to full-semantic verbs with a typical object (***roníme*** *slzy = we* ***shed*** *tears*).

As we depend on the UD annotation, the most straightforward decision on choosing verbs is to check the upostag of the token and respect the distinction between the two categories: include all words with the *VERB* upostag into the dictionary and leave all words with *AUX* upostag outside. This would approximately lead to the exclusion of auxiliary verbs (2.1.3) and copulas (2.1.4) on one side and the inclusion of phase verbs (2.1.6) and light verbs (2.1.7) on the other side .

The case of modal verbs is more complicated, as their annotation is not consistent among the languages (e.g. in English they are marked as auxiliaries, in Czech as verbs). We must decide on whether we want to include them into the dictionary building valency frames for them and, in the case of both answers, how to include or exclude modal verbs for languages that mark them the opposite way we need. The issue of modal verbs is examined in more detail below in 2.1.5.

Finally, there are also special cases of fixed forms of older verbs or of nominals which behave like verbs (Czech *lze*, meaning *it is possible* or Ukrainian треба */treba/*, meaning *it is needed*) and might be annotated in different ways. However, these cases are few, so their potential mistaken inclusion or exclusion into the dictionary is not treated, all the more it not clear, which option we should consider correct.

2.1.3 Auxiliary verbs

Auxiliary verbs are verbs without their own meaning, serving only to participate in compound forms of other verbs. In Czech such an auxiliary verb is *být* (*to be*) which is used to form the future tense (4), past tense (5) (6), passive voice (7) and conditional mood (8). In English, they are e.g. *to be* for continuous tenses (5) and the passive voice (7), *to have* for perfect tenses (6) and *to do* for polar questions (9) and negations (10). These verbs should bear *AUX* upostag and depend on the full-meaning verb with an *aux* deprel. They are not appropriate for inclusion into the valency dictionary, because they are only formally independent, functionally they are actually just a part of the full-meaning verb they are used with and as such they do not have their own valency frame. However, they may still be used in a full-meaning sense, of course (11) (12) (13).

(4) "*budu* psát" = "I will write"

(5) "psal *jsem*" = "I wrote" / "I *was* writing"

(6) "napsal *jsem*" = "I wrote" / "I *have* written"

(7) "*jsem* napsán" = "I *am* written"

(8) "napsal *bych*" = "I would write"

(9) "*do* you write?"

(10) "I *do* not write"

(11) "Bůh *je*." = lit. "The_God is.", meaning "The God exists."

(12) "I *have* an apple."

(13) "You *did* a good job."

2.1.4 Copulae

Copulae are specific auxiliary verbs introducing nonverbal (mostly nominal) predicates. In most cases the verb *to be* serves as the copula (14) (15). In many languages the copula is completely omitted, at least in the present tense (16). The upostag of copulae are also *AUX*, but they take a special *cop* deprel. Sometimes also other verbs that *to be* can be traditionally perceived as copulae (17), but not in UD. Like other auxiliary verbs, copulae should not be included into the valency dictionary.

(14) "My mom **is** a teacher.", "I **am** tall.", "They **were** in the garden."

(15) "Moje máma **je** učitelkou.", "**Jsem** vysoký.", "**Byli** na zahradě."

(16) "Моя мама учителька." /moja mama učyteľka/ lit. "My mom teacher", "Я високий." /ja vysokyj/ lit. "I tall.",
"Вони були у саді." /vony buly u sadi/ lit. "They **were** in garden." (UA)

(17) "Petr se **stal** vítězem." = "Petr **became** the winner."

2.1.5 Modal verbs

Modal verbs are a category of verbs that build a compound predicate with another verb and express several attitudes, that the subject can have towards the action performed, such us capability, possibility, permission, necessity, willingness etc. Modal verbs are not autosemantic, because they usually accompany another verb expressing the main action, that is only modified by the modal meaning. On the other hand, they are not completely auxiliary as they have their own semantic meaning unlike purely grammatical auxiliary verbs (e.g. *I **have** said* or *we **are** sitting*).

Their annotation differ across languages: sometimes they are considered verbs, annotated with *VERB* upostag (e.g. in Czech), other time they are marked *AUX* as auxiliaries (e.g. in English). Their position in the dependency tree also varies. As auxiliary verbs, they are attached under the full-meaning verb, which is more semantic-based approach. If they are considered usual verbs, they are the head of the phrase with the full-meaning verb as their argument, which is more syntactic-oriented way, because it is usually the modal verb, that is in the finite form and agrees in various grammatical categories with the subject. The approaches varies also in attaching other arguments to the predicate. The subject is dependent always on the head of the phrase, whereas other arguments, objects and obliques, are dependent on the full-meaning verb. This must be taken into consideration when extracting the valency frame of both, the modal verb (if we decide to extract it) and the full-meaning verb. The different approaches are demonstrated on the same sentence in English and Czech in Fig. 2.1. Eventually, there are languages, where the modality may be expressed by other means, e.g. verbal affixes (e.g. Turkish: *çizerim = I draw*, *çizebilirim = I can draw*, *çizmeliyim = I must draw*) or non-verb auxliary verbs (e.g. Ukrainian adverb треба */treba/* denoting need, as in мені треба іти */meňi treba ity/ = I need to go.*

It is necessary to decide, whether we want to include the modal verbs into the dictionary or not. Both cases would mean their explicit, language-specific

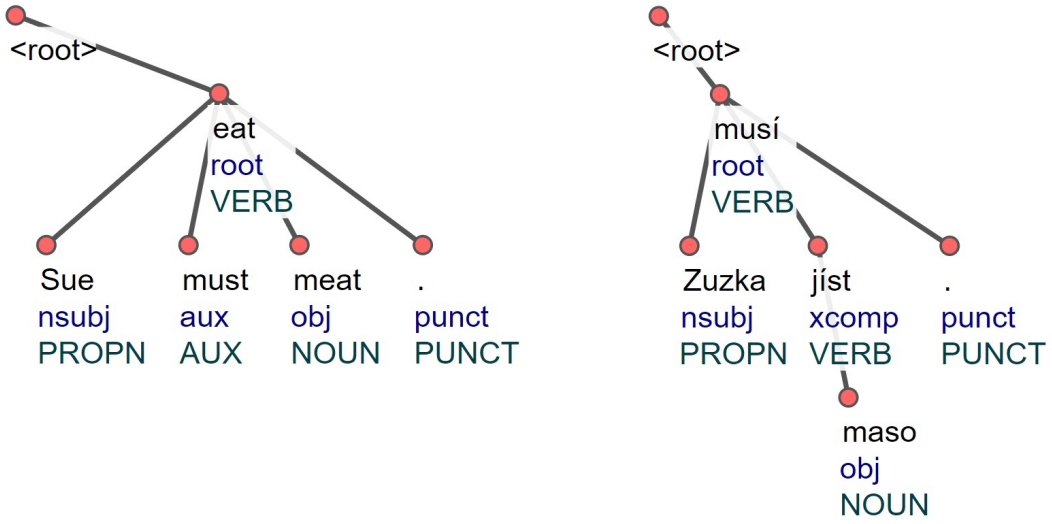


Figure 2.1: Different approaches to annotation of modal verbs.

inclusion or exclusion in the languages that normally annotate them the other way. However, both directions would require defining a list of modal verbs for the languages that follow another annotation decision that our chosen one. The list depend strongly on particular language tradition and even after the addition or removal of the modal words from such list, some corresponding verbs may be treated differently during the bilingual valency frame extraction. The modal verbs are not a marginal matter, they comprise about 7 % of all verb occurrences, so it is important to choose a good solution. I did not take a definitive decision in this question. Instead, each language-specific extractor specifies the list of modal verbs according to the language tradition and specifies also, if the modal verbs in UD annotation are marked as verbs or as auxiliaries. Accordingly, each extractor contains an extraction unit for the opposite treatment with the modal verbs. If it excludes them by default, it has a unit for their explicit inclusion and vice versa. Unlike other extraction units of the extending extractors, these modal verbs units are deactivated by default and can be explicitly activated in the configuration file E. The units dealing with the modal verbs including the mentioned lists are described in 2.5.2 for Czech and Slovak and in 2.6.2 for English.

2.1.6 Phase verbs

Phase verbs describe a state of the action or phenomenon described in the predicate, like starting, continuing or stopping (18). Despite the phase seems to be similar to the modality in description of the circumstances, the phase verbs are more often perceived as common verbs, which just usually take a verbal object (19). So do the UD, where the phase verbs are annotated with *VERB* upostag and *xcomp* deprel. This leads to their inclusion into the dictionary, which is desired, because it allows us to compare different frames of these verbs (20).

(18) "Přestala jíst$_{[xcomp]}$" = "She stopped eating$_{[xcomp]}$."

(19) "Slíbila jíst$_{[xcomp]}$" = "She promised to eat$_{[xcomp]}$."

(20) "Martin začal práci*[obj]*." = "Martin started the work*[obj]*."
"Martin začal pracovat*[xcomp]*." = "Martin started to work*[xcomp]*."
"Začalo pršet*[csubj]*." = "It stared to rain*[csubj]*."

2.1.7 Light verbs

Another type of non-full-meaning verbs are constructions known as light verbs, which are common verbs used in a compound predicate with a typical (mostly nominal) complement. The meaning expressed can be quite distant from both the verb and its complement. For example a Persian verb (indo-iranian languages are well known for a rich use of light verbs) هرف زدن (*harf zadan*) meaning *to talk* or *to chatter* can be literally translated as *to beat letters*. The distinction between light verbs with their typical complement and full-meaning transitive verbs with an often occurring object may be not only semantic, but also syntactic. Often the whole phrase of light verbs with its complement acts as transitive verb and takes another object (21). But in most cases, it is very difficult to distinguish light verbs from full-meaning verbs, the border between these groups is often unclear.

The meaning expressed by such phrase in one language is often expressed by a separate verb in another language: *take a nap* (in Czech *zdřímnout si*). The light verb works here as a bearer of grammatical categories of the predicate while its semantics is mostly expressed by the nominal part. Moreover there is no cross-lingual agreement on the annotation of the light verbs in UD. There are languages, where light verbs are indicated by a compound deprel (*compound:lvc* - light verb construction) of the complement depending on the verb. However, this does not hold for all languages, where the the nominal part of such light predicate is perceived as a common object (22). For the stated reasons, it would be hard to treat the light verbs properly. The decision I took does not consider *compound* dependents and treats all objects equally. Thus in both cases only the verbal part is considered, but the nominal part is in one case omitted, in the other case considered a usual object. This is not an ideal treatment of light verbs and there is still space for improving the program so that it considers at least the light verbs marked with *compound* dependents. On the other side, there is no way to distinguish light verbs with common transitive verbs for languages, where the nominal part of the light verbs is marked as *obj*.

(21) "کار را شروع کردم" ("kār rā šorū' kardam") (FA)
= lit. "the job start I_did", meaning *I started the job*

(22) "به چرتی زدم." ("be čortī*[compound:lvc]* zadam.") (FA)
= lit. "to a_nap I_hit.", meaning "I took a nap*[obj]*."

2.1.8 Verbs with verbal particles

Among the languages, there are some other groups of verbs that are usually used together with another word. I will call these accompanying words *verbal particles* for now, as I believe there is no common term for them. Although the issue of considering the verb particles during the frame extraction belongs rather to the section about argument extraction, the verbs with verbal particles have something

in common with light verbs (a separate non-verbal part modifying the proper meaning of the verb), so I describe them here. Let us list three types of them as examples, but certainly there are other occurrences in other languages. Verbs using verbal particles are often translated into other languages with a single word (i.e. a verb without any verbal particle), which is always shown in the example sentences.

The first case are phrasal verbs in English. They are verbs used with so called phrasal particles, which are originally adverbs (*calm down*, *figure out*) or prepositions (*hold on*, *look for*). Omitting the particle in most cases changes the meaning of the verb a lot (*take off* vs. *take*). In UD, the phrasal particles are marked with *compound:prt* deprel (23).

(23) "The plane takes off*[compound:prt]*." = "Letadlo vzlétá."

Secondly, in several languages, there are morphemes that under some conditions are a part of the verb token and in other situations form a separate word. This is also the case of German separable prefixes (24), annotated in UD the same way as English phrasal particles, or Hungarian preverbs (25), annotated with a *compound:preverb* deprel. The conditions for the separation differ: in German, they are separated in finite forms in main clauses, while in Hungarian, the separation is imposed by negation or a question.

(24) "Ich hole dich morgen **ab***[compound:prt]*." (DE)
= "I will pick you **up***[compound:prt]* tomorrow."
= "Vyzvednu tě zítra."
"Ich kann dich morgen **ab**holen." (DE)
= "I can pick you **up***[compound:prt]* tomorrow."
= "Můžu tě vyzvednout zítra."

(25) "A hangyák **ki**másznak az utcára." (HU)
= "The ants are crawling **out** into the street."
= "Mravenci vylézají na ulici."
"A hangyák nem másznak **ki***[compound:preverb]* az utcára." (HU)
= "The ants are not crawling **out** into the street."
= "Mravenci nevylézají na ulici."

Yet another example of verbal particles are reflexive pronouns in inherently reflexive verbs used e.g. in various Romance and Slavic languages. These verbs cannot be used without the reflexive pronoun (although they probably could be used so in the past), so it becomes redundant. This case is different from the English phrasal verbs, which can be used also without the phrasal particle (although with a change in the meaning. The use of an inherently reflexive verb without the reflexive pronoun would be grammatically incorrect. The reflexive pronoun works as a formal object (in Czech it is in accusative (26) or dative (27)) and the true object (if the verb is transitive) is in an indirect form (in an non-typical case for the objects, e.g. genitive (26), or with a preposition (27)). The obligatory reflexive pronouns are denoted with a *expl:pv* deprel in UD (see also expletive in 1.3.7). Of course, there are also true transitive verbs in Czech and Spanish, which can be used apart from the reflexive pronouns also with any

other object. Note that in some other Slavic languages, the reflexive pronouns (including the cases of the inherently reflexive verbs) are attached to the verb as its suffix, which eliminates the need of their special treatment when extracting valency frames (28).

(26) "Bojím **se**$_{[ACC,expl:pv]}$ vlků$_{[GEN]}$."
= "I am afraid of wolves."

(27) "Se$_{[expl:pv]}$ queja de mí." (ES)
= "Stěžuje si$_{[DAT,expl:pv]}$ na mě."
= "She complains about me."

(28) "Я боюся я вовків." /ja boju**sja** vovkiv/ (UA)
= "I am afraid of wolves."

Ideally, the verb particles of all three types (phrasal verbs, verbs with separable affixes, inherently reflexive verbs) should be a part of the verb lemma in the dictionary. Marked as expletives, the reflexive pronouns of inherently reflexive verbs are extracted as normal arguments. The separable affixes lead to the creation of two analogous valency frames with identical arguments, differing in the presence of the affix in the verb lemma. Language-specific extractors should treat the separated affixes and add them explicitly to the frame. The phrasal verb particles are not included automatically and their inclusion is also left to the language-specific extractors, which is actually implemented in the English extractor in this work (see 2.6.7).

2.2 Selection of arguments

This section discusses the selection of valency arguments. After explaining how the process is generally done (2.2.1 - 2.2.2), more detailed elaboration of the criteria for choosing an argument and their application to complements with various deprels takes place (2.2.3 - 2.2.8). Finally the extraction of various attributes of the chosen arguments is described (2.2.9 - 2.2.10).

2.2.1 UD annotation relevant for argument selection

. After the verb as the core of a valency frame has been selected, it comes the time for the second part of the extraction - choosing its frame arguments. It makes sense to assume that the arguments should be contained among the children of the verb in the UD tree structure of the sentence, although various language-specific transformations or technical solutions not corresponding to our concept of valency frame may cause that an argument is in another place in the tree: it may be e.g. the parent of the verb or its sibling like in the Fig. 2.2. Apart from the tree structure we work with other information contained in the UD annotation like upostags, relevant features and deprels. Vice versa, not all the children of the verb should be included among the frame arguments.

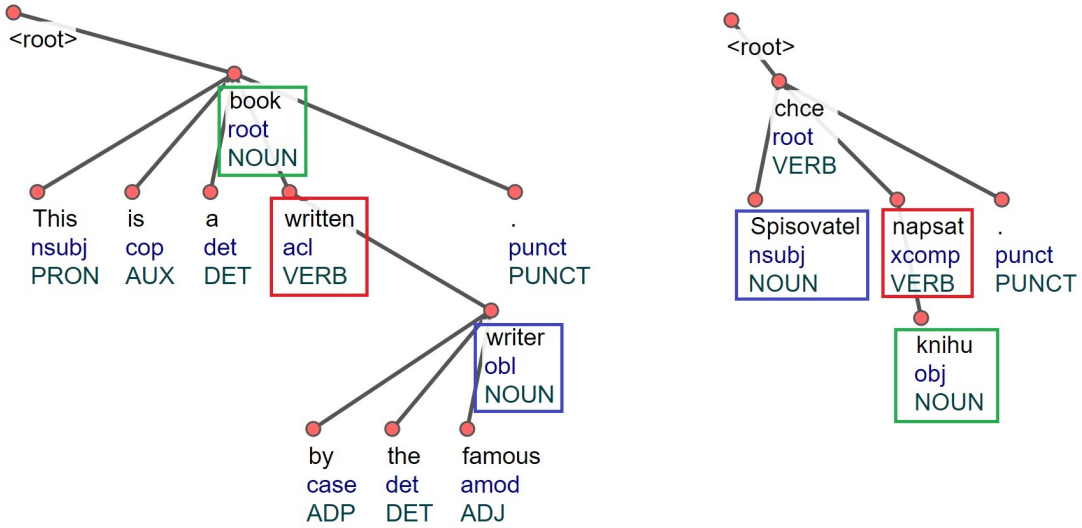


Figure 2.2: Examples of sentences, where the arguments (subject in blue, object in green) of the verb (in red) are not always its child. Both cases are non-standard. In the English sentence "This is a book written by the famous writer." the verb is a passive participle, the Czech sentence "Spisovatel chce napsat knihu." = "The writer wants to write a book." includes a modal verb.

2.2.2 Argument selection algorithm

The selection of arguments has two parts: choosing the arguments themselves and finding the defining set of their attributes (which are an important part of the output dictionary for the user, but also serve to compare arguments in order to say whether two frames are identical - see above in 2.0.7). The algorithm iterates through the child nodes of the verbs and for each of them decides, whether they should be included. If yes, it extracts all its relevant attributes. In fact, the choice of the argument depends on its attributes: when the algorithm has all the necessary characteristics of the verb's dependant, it can decide whether to consider it a valency argument or not. The basic criterion is the deprel, but language-specific extractors may take also other properties into consideration.

2.2.3 Criteria for argument selection in FGD

In order to decide, which deprels should lead to argument inclusion, let us remind the definition of the valency frame in FGD (1.2.3), which is the main inspiration for the notion of valency frame used in this work. According to it we should select obligatory actants, optional actants and obligatory adjuncts. This would involve developing a program with the ability to do the two distinctions, based on the information contained in UD annotation: between actants and adjuncts and between obligatory and optional complements. We will see that the UD annotation does not allow reliably to do this in general.

2.2.4 Actants vs. adjuncts

As it was already discussed in 1.3 the actants in FGD roughly correspond to the notion of core in UD. More precisely, the core seems to be a subset of actants as

all the core arguments (*nsubj*, *obj*, *iobj*) and their respective clausal counterparts (*csubj*, *ccomp* and *xcomp*, see Tab. 1.4) appear to be actants. The deprels *expl* and *obl* deserve more detailed commentary and are discussed in the following paragraphs. Relations like *advmod* or *advcl* map clearly to the adjuncts. The rest of deprels discussed before (*aux*, *cop*, *mark*, *compound*) are function words forming a part of the predicate itself or describing its relation to its head, so they are not part of its valency frame.

2.2.5 Expletives

. Although expletives are semantically empty words, they are still at least formally relevant for the valency frame of the verb and they describe how the verb is used. Moreover, they can be important for the differences between individual valency frames. Let us compare the standard use of the verb *to rain* (29) with a little bizarre dialogue between rainclouds (30). In the first case, there is an avalent verb with an expletive subject, in the second case, we have a monovalent verb with a proper subject coreferring with a previously mentioned entity. These arguments lead us to the decision to include the expletives into the extracted valency frames, probably with a special mark.

(29) "It is raining outside, we cannot play football."

(30) "'What is our child doing, honey?' asked the daddy raincloud his wife."
"'It is raining,' she answered."

2.2.6 Obliques

As described in the previous section, our valency frames aim to contain all actants, which means also part of the oblique dependants. This is a difficult task, because obliques can represent both actants and adjuncts and there is no reliable means to determine which occurrence of an oblique belongs to which type. Unlike other complements, that are included into the frame or left out of it solely based on their deprel, the inclusion or exclusion of obliques requires further analysis and implementation of heuristics. That is non-trivial, so it is part of the main extractor (see 2.4.5).

In the base extractor, where only linguistically straightforward solutions are used, are oblique complements always excluded by default. I gave preference to this option over the opposite one, because the role of obliques as adjuncts is more frequent than actants (as shown later in Tab. 2.12).

2.2.7 Obligatory vs. optional

Regarding the second distinction, between the obligatory and optional complements, the situation is more complicated, because in the annotation itself, no such information is contained. There is a possibility to try to find out the obligatoriness of complements from the data. The idea would be following: if a complement occurs with the verb often, it is rather obligatory, if it occurs rarely, it is rather optional. This criterion is obviously risky, because the obligatory complements can be in some cases unexpressed, if they are understood from the context, and

in contrast the optional complements can be very frequent and still be optional (Vallex has a separate category for this types of complements: typical). There are no means to avoid this fusion without deeper semantic annotation of the data.

The classification of the complements according to their obligatoriness is important especially for the adjuncts, because that decides, whether they should be included into the valency frame in FGD terms. The actants are included always, regardless of their obligatoriness. Because the possibility of misclassifying the complements regarding their obligatoriness is quite big and the contamination of the extracted valency frames with many optional adjuncts would cause a large harm to the resulting dictionary, I decided not to include adjuncts into the extracted valency frames. On one hand, this is a deviation from the FGD concept of the valency frame and the extracted valency frames would say less information about the verb. On the other hand, the dictionary should be purer and more reliable in what it aimed at.

2.2.8 Deprels leading to argument inclusion

To sum it up, the presented program aims at including only actants (both obligatory and optional) into the valency frame as arguments and does so for following deprels: *nsubj*, *csubj*, *obj*, *iobj*, *ccomp*, *xcomp*, *expl* and sometimes also *obl*.

2.2.9 Selection of argument attributes

The defining attributes of the arguments should describe their form and function in the sentence (examples of these are shown in the sentence (31) under letters A - F, depicted also in the Fig. 2.3). The function is the argument's dependency relation to the verb which is described by its *deprel* (B and E). The representation of the form depends on the part-of-speech of the argument. The form of a nominal is represented by its morphological case, denoted by the *Case* feature (C), and its adposition (preposition or postposition), which is from the point of view of valency syntax considered a similar phenomenon and is marked with the *case* deprel (A) (see also its description in 1.3.12). In the case of verbs, i.e. subordinate clauses that are an argument of the head verb of the frame, their form is found in the *VerbForm* feature (F). There may be also a marker (typically a conjunction) connecting the clause to the head verb, which is also considered an important feature of the argument. The dependency relation of the marker to the argument is *mark* (D). To sum up, the defining attributes of a valency argument are: its deprel, its *Case* and *VerbForm* features and lemmas of its *case* and *mark* dependants.

(31) "Zjistím*[VERB]* od*[A - case]* bratra*[B - obl:arg][C - Case=Acc]*, zda*[D - mark]* zpívá*[E - ccomp][F - VerbForm=Fin]*."

The adpositions and markers depend are separate nodes depending on the argument in UD, so the algorithm should look even at the child nodes of the arguments (i.e. grandchildren of the head verb). These descriptions may be combined in various ways (e.g. verb forms like participles may have a case or an adposition (32). Finally, an argument in some languages or under various circumstances may also completely lack any description of the form.

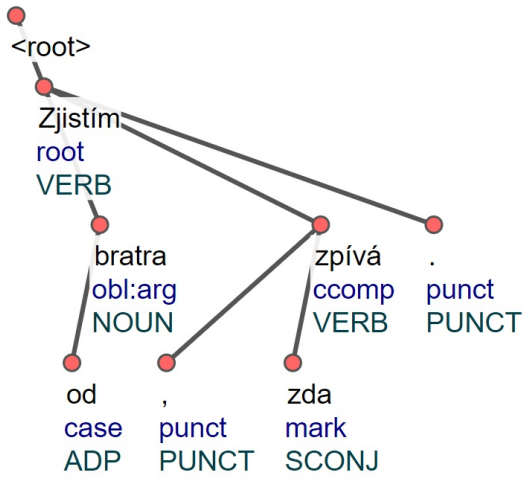


Figure 2.3: Dependency tree of the example sentence (31).

(32) "Prohlásili*[VERB]* ho za*[case]* šíleného*[obl:arg, VerForm=Part, Case=Acc]*." = "They declared him crazy."

2.2.10 Absence of semantic roles

Many valency theories including some of the discussed ones do not end at the choosing the arguments for the valency frame, but they also include assigning semantic roles to them (Fillmore calls them semantic cases, in FGD they are named functors). The inventory of the semantic roles differs among the theories both in their number and their names. In FGD, the basic functors are agent (actor) and patient, which usually correspond to the syntactic terms subject and object. However, the point of the passive diathesis is making the patient act as the subject. Beside of it, many intransitive verbs describe a state of its subject rather than action and in this case may seem inappropriate to assign the agent label to it (e.g. *to sleep*), although FGD does so. Moreover there are transitive verbs, where it is not clear which argument should be labeled with which semantic role and the arguments have different syntactic functions among the languages, e.g. the English verb *to like* vs. the inherently reflexive Czech verb *líbit se* (33) (although in this particular case, the reflexiveness and the use of dative instead of accusative in the case of the Czech verb would lead us consider the English object and the Czech subject as patient and the other argument as the agent). Except of the mentioned examples, it would not be a big issue to assign the agent role to the subjects a the patient role to the objects, which would be correct in most cases, but the assignment of other semantic roles (like addressee, effect and origin in FGD) would be hard, since UD lacks information necessary for finding the semantic function of all the oblique dependents. This lead me to the decision not to assign any semantic roles to the extracted valency frame arguments. The arguments are described only by their required morphological form and syntactic relation to the verb.

(33) "Pavlovi*[DAT,obj:arg]* se líbí Alice*[NOM,nsubj]*".
= "Paul*[nsubj]* likes Alice*[obj]*".

2.3 Extending frame extractors

The basic extractor described in 2.1 and 2.2 (and implemented in `Base_frame_extractor` class) contains mostly technical core of the extraction algorithm choosing the simplest, trivial linguistic solutions and it can be extended by more advanced extractors (as mentioned in 2.0.6). Several such extending frame extractors are presented in following sections (2.4 - 2.6). In this section some important points common for all of them are shortly described, like their language-universality or language-specificity (2.3.1), their implementation (2.3.2 - 2.3.3) and instructions for interpreting their evaluation shown for each of them (2.3.4).

2.3.1 Main extension vs. language-specific extensions

The basic algorithm chooses trivial solutions and is language-independent, of course, but that does not mean that all extending extractors must be language-specific. The first presented extending extractor, the main extractor (implemented in `Frame_extractor` class), described in see 2.4, is still language-independent and implements a non-trivial treatment of several universal language phenomena improving the extraction in comparison with the basic extractor. The language-specific extractors are based on it and further improve the extraction process for particular languages or language groups. These extractors are presented in 2.5 and 2.6.

2.3.2 Implementation of extending extractors

The extension is always realized as class inheritance, where the extending extractor overloads methods of its parent extractor or adds new ones. The language-specific extractors can be inherited directly from the main extractor or have a more complex inheritance structure. Extractors for three languages are provided in this work: the English extractor 2.6 is a direct descendant class (`En_frame_extractor`) of the main extractor, whereas the Czech and Slovak extractors are commonly inherited from a Czech-Slovak extractor 2.5 (`Csk_frame_extractor` class), which is inherited from the main one, because most of the extraction process is the same for both languages and the two specific extractors (`Cs_frame_extractor` and `Sk_frame_extractor` classes) add only minor changes (Fig. 2.4). Other language-specific extractors, that might be added in the future, should be ultimately inherited from the main extractor, but can also have a more complex inheritance structure, like Czech and Slovak extractors.

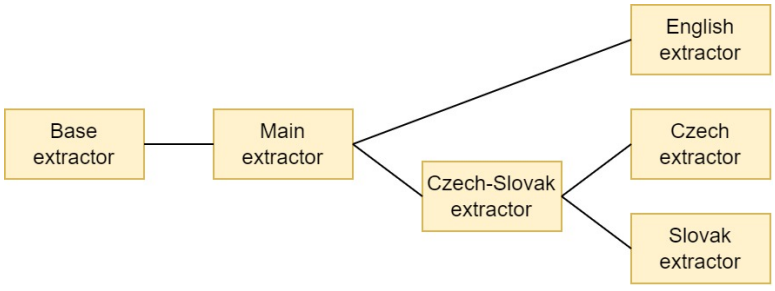


Figure 2.4: Overview of the inheritance structure of the provided extractors.

2.3.3 Extraction units

Extending extractors (the main one and the language-specific ones) always consist of several extraction units, each dealing with one language phenomenon. Some units work on argument level, influencing the selection of arguments and their description for a given verb (2.2), others work the frame level, choosing the verb itself (2.1) and some may work on both levels. Several units do postprocessing with the frames after the whole treebank is processed. The work of one unit may influences the others. The units can be individually activated or deactivated (see E), with each having a default setting (mostly activated). Each extending extractor is constructed in such way, that deactivating all its units should give the same extractor and the same results as the original parent extractor, on which the extension is based. The main extractor adds some units to the base extractor, the language-specific extractors have add even more units specific for the particular language.

2.3.4 Description and evaluation of individual extraction units

In the following sections, all extending extractors are individually covered, each by its units. For each extractor, the frame-level units are presented first, the argument-level units after them. Every unit is described and related statistics demonstrating the frequency of the associated phenomena are shown, which are measured on Czech and English PUD.

The evaluation of the units is presented together in the evaluation section (see 2.7 in order to compare the units mutually. The description of the evaluation data, process and metrics can be also found there. However, it is useful to be shown individually also here, after the description of each unit so I point out several most important things about evaluation here. Unlike the statistics mentioned, which are measured on all 1000 sentences in PUD, the evaluation is performed only on a its subset of 100 sentences. More about the evaluation data can be found in 2.7.1. For every unit two tests are performed, each with two extractors (they are called BASE, PLUS, MINUS, FULL): an extractor with the unit (PLUS, FULL) is evaluated against the other one without it (BASE, MINUS). This process is explained in more precisely in 2.7.7. In each test four or two values are measured: four for frame-level units and two for argument-level units, for which the first two values are always identical for both extractors in the test, because the use of the unit does not affect the value. These values are called *frame ID*, *lemmas*, *arg ID* and *arg desc* and are described in detail in 2.7.8. Finally, the improvements between the two extractors in the test (from BASE to PLUS and from MINUS to FULL) are measured for each value, which is further elaborated in 2.7.9.

Several extraction units are deactivated by default, so to reproduce their test, they must be activated in the configuration file first (see E). These units are not included in the tests of other units, so the results for the FULL extractor differ from the FULL extractor results given fot other extraction units.

2.4 Main extension

The main extractor extends the basic extractor and adds several more linguistically based language-independent extractions units. It deals with missing subjects 2.4.1, form of compound verbal arguments 2.4.2, coordination 2.4.3, obliques 2.4.4 - 2.4.5 and frame reduction 2.4.6.

2.4.1 Subject adding (`subj`)

An important modification above the simple extraction of the listed deprels is addition of subjects. This is based on an approximate assumption that every valency frame has a subject. This is useful in many situations, where the subject is missing and the frame would be extracted without it. In subject-dropping languages (e.g. Czech), the subject pronouns can be used for emphasis, but usually they are elided and the person and number they express are understood from the verb form. But there are also various infinite forms of verbs, such as infinitives, gerunds or participles, that do not have a subject expressed in the sentence (34) (35) or at least it is hard to determine it reliably, as it may be one of various arguments of a superordinate verb (36) (37). Therefore the subject is added to the frame as elided even if it is actually present somewhere in the sentence.

(34) "Řídit$_{[VERB]}$ autobus*obj* v noci$_{[obl]}$ je nebezpečné."
= "Driving$_{[VERB]}$ a bus*obj* at night$_{[obl]}$ is dangerous."

(35) "Král nařídil opravit$_{[VERB]}$ hradby$_{[obj]}$."
= "The king ordered to repair$_{[VERB]}$ the walls$_{[obj]}$."

(36) "Zedník$_{[nsubj]}$ slíbil králi opravit$_{[VERB]}$ hradby$_{[obj]}$."
= "The bricklayer$_{[nsubj]}$ promised the king to repair$_{[VERB]}$ the walls$_{[obj]}$."

(37) "Král nařídil zedníkovi$_{[nsubj]}$ opravit$_{[VERB]}$ hradby$_{[obj]}$."
= "The king ordered the bricklayer$_{[nsubj]}$ to repair$_{[VERB]}$ the walls$_{[obj]}$."

The subjects are added with *nsubj* deprel, although in many cases *csubj* would be also possible. However, any further information about their form is left to the language-specific extractors (see 2.5 and 2.6). The verbs may also miss another arguments, like objects, but it is not possible to add them universally, because they could be incorrectly added to a intransitive verb.

Actually, the mentioned assumption about each verb having a subject does not hold universally as well, because there are avalent verbs, that miss any arguments, especially the subject (or at least some of their frames do). These verbs are usually used in the third person singular and semantically they often describe several weather phenomena, like *prší = it is raining* or *sněží = it is snowing*. These verbs typically have also a frame requiring a subject in case they are applied (often as a metaphor) to different substance than usual, i.e. the rain or the snow, even to a different person as in lyrics of a Czech song(38).[1] There are also other such subjectless constructions and interpretation of some of them depends on other frame arguments (39) and it would not be easy to list them all and

[1] *Obchodník s deštěm* by the music band *Kryštof*.

	CS		EN	
	PLUS	FULL	PLUS	FULL
# of frames	1755		2153	
all forms %	40.5	35.2	37.1	32.9
- finite %	9.3	7.5	3.6	1.1
- infinitive %	15.8		12.6	11.9
- participle %	15.2	11.7	8.7	7.8
- gerund %	-		12.1	12.0
- other %	0.2		0.1	

Table 2.1: Statistics of the subject adding unit: how many subjects were added to a verb in which form.

CS	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	79.11	94.58	74.05	79.13	94.43	73.31
arg descr	83.36	69.89	-80.95	85.59	72.6	-90.15
EN	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	69.47	84.16	48.12	72.87	87.39	53.52
arg descr	45.36	39.93	-9.94	47.74	42.01	-10.96

Table 2.2: Evaluation of the subject adding unit on the Czech and English data.

distinguish their frames with and without subjects. The chosen solution is to add the unexpressed subject to the frame in all cases, because the cases of avalent frames are rare and the error caused by this solution would be small.

(38) "Slova jsou jen kapky deště a ty voláš, ať prším ještě."
= "Words are just raindrops and you call, so that I rain more."

(39) "Jde o tebe" = lit. "It_goes about you.", meaning "It is about you."

On the Tab. 2.1, we can see counts of verb frames, to which the subject was added by this unit. This happened mostly for various non-finite verb forms lie infinitives, participles or gerunds, especially in English, where finite forms usually must have a subject. The table also shows that with combination with other units, slightly less subjects are added, which is because of the coordination unit that adds the subjects from the first frame of the coordinated structure to the others, where it missing. The contribution of the subject adding unit is shown in the Tab. 2.2.The improvement of argument selection is very high, because the gold frames all have subjects. Improvements of the argument description are negative, because the description of form (i.e. nominative for Czech and English) is left on the left on the language-specific extractors (see 2.5.5 and 2.6.3) so the added subjects are missing a form which worsens the results.

2.4.2 Auxiliary finite form preference (`auxf`)

Clausal arguments, like nominal arguments, are extracted with their form (finite form, infinitive, participle, gerund...) and introducing words (typically conjunctions or adverbs). If the clausal argument is a complex structure consisting of a full-meaning verb with an auxiliary verb, it is often the auxiliary verbs that bears the proper finite form, required by the frame, whereas the full-meaning verb is in different form than it will be in the situation without the auxiliary verb, typically in a non-finite form. But because one of the principles of UD is the semantic priority, simple argument extraction of the upper node form would lead to two separate frames for the main verb (for which this clausal argument is extracted): one with the argument in finite form from the situations, where the full-meaning verb stands alone, another with the non-finite form from the cases, with the auxiliary verb taking over the finite form. To prevent this, there is a modification, which in the case of a clausal argument with an auxiliary verb in a finite form prefers the the finite form of the auxiliary verb to the form of the full-meaning verb itself (40). A similar situation occurs in the case of nominal clauses with a finite-form copula, where the copula's form is preferred to the nominal form (41). If the clausal argument has a non-finite form and it there no auxiliary verb or none of its auxiliaries has a finite form, it keeps its original, non-finite form.

(40) "Ptá se, co děláš*[Fin]*." = "She asks, what do you do."
→ [co + Fin]
"Ptá se, co budeš*[Fin]* dělat*[Inf]*." = "She asks, what will you do."
→ ~~[co + Inf]~~ [co + Fin]

(41) "Říká, že tancuješ*[Fin]*." = "He says that you dance."
→ [že + Fin]
"Říká, že jsi*[Fin]* učitel*[Nom]*." = "He says that you are a teacher."
→ ~~[že + Nom]~~ [že + Fin]

Tab. 2.3 shows counts and portions of clausal arguments that were already originally finite, changed to finite by the auxiliary form preference unit or left in an originally non-finite form. Other units have almost no impact on the work of this unit (so the counts are the in most cases similar for both test runs). The majority of the clausal arguments kept its original form, only few of them had their form changed to finite. The contribution of this unit is shown in the Tab. 2.4. The unit does not affect argument selection, only changes the form of some arguments. We can see that there is a small improvement in Czech and almost none in English.

2.4.3 Coordination (`coor`)

Another problem worth of mentioning is processing of coordinated predicates, which may share some of the arguments. Such arguments are dependent on the coordination head, which is always the first conjunct (see 1.3.14). Therefore it might not be clear whether the arguments of the heading conjunct hold exclusively for it or also for other conjuncts in the coordination structure. On the other hand, the arguments dependent on the other conjuncts are certainly only theirs. Counts

	CS		EN	
	PLUS	FULL	PLUS	FULL
# of args	2760	3680	2636	3735
clausal args %	16.0	12.0	14.2	10.1
- originally finite %	2.5	1.9	1.4	1.1
- changed to finite %	2.5	1.8	3.1	2.2
- left non-finite %	11.1	8.3	9.7	6.9

Table 2.3: Statistics of the auxiliary form preference unit: how the form of clausal arguments was treated.

CS	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	79.11	79.11	0.0	94.43	94.43	0.0
arg descr	83.36	84.98	9.74	71.26	72.6	4.66
EN	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	69.47	69.47	0.0	87.39	87.39	0.0
arg descr	45.36	45.48	0.22	41.89	42.01	0.21

Table 2.4: Evaluation of the auxiliary finite form preference unit on the Czech and English data.

of coordinated predicates are shown in the Tab. 2.5.[2]

	CS	**EN**
# of all verbs	1755	2153
2 coordinated %	12.3	8.2
3 coordinated %	0.7	0.5
4 coordinated %	0.1	0.0

Table 2.5: Statistics of the coordination unit: numbers of coordinated predicates with 2, 3 or 4 verbs.

The problem has no reliable solution, the situations can be interpreted ambiguously: e.g. in (42) the word *songs* can be the object of both verbs or only of the second one. One heuristics that can be used is based on the observation that common arguments behave to the whole coordination structure as to one predicate, so they either precede or follow it completely, so arguments between the predicates (i.e. in the middle of the structure) are not shared, as in (43), where the word *poems* is certainly dependent only on the first verb.

(42) "Peter sings and composes songs."

(43) "Susan writes poems and sings songs."

[2]This statistic can be obtained running the `treebank_examiner.sh` script.

	CS		EN	
	PLUS	FULL	PLUS	FULL
# of args	2856	3680	2729	3735
possible coord args %	4.9	5.6	3.7	3.5
deleted coord args %	1.4	2.1	0.3	0.9

Table 2.6: Statistics of the coordination unit: number of arguments marked as possibly incorrect due to coordination of predicates and number and portion of those truly deleted by the simple heuristic.

Regarding the rest of the arguments that cannot be decided by the mentioned heuristics, they are decided with respect to the other occurrences of the verb. The frames of the non-heading conjuncts are extracted with all the arguments of the heading conjunct included, but marked with a marked saying they were taken due to coordination and thus their inclusion is not definitive. After the whole treebank is processed, the frames with such marked arguments are considered for removing the arguments. The easiest decision is used in this work: if the argument concerned is marked as taken from coordination in all occurrences of the frame, it is removed (and the frame might be merged with another frame subsequently), if there is an occurrences of the frame with a proper occurrence of the argument concerned, the argument remains in the frame. The numbers of the all selected possible arguments due to coordination and of the later deleted arguments ones are shown in Tab. 2.6.

The algorithm could be improved in several ways. There can be a threshold requiring a minimal portion of the frame occurrences with the argument not taken from coordination (i.e. not only one such occurrence). Even better option would be look at the number of occurrences of the corresponding frame without the argument concerned. If there were many occurrences without the argument, it would support the removal of the argument, if there was substantially more occurrences with the argument, it would lead rather to preserving the argument. Yet another possible heuristic could be measuring the distance between the two predicates, because by observation we can say that verbs sharing arguments are often close to each other, whereas when there are long, complex dependent structures between them (such structures themselves are not taken into consideration by the first, used heuristic), it lowers the chance that the surrounding (especially following) arguments are shared. This is because in the case of shared arguments, people need to perceive the coordinated verbs as one unit, which is more difficult when they are far from each other.

The coordination unit shows only small improvement, more apparent only in argument selection for English (see Tab. 2.7).

2.4.4 Oblique dependents (`oblq`)

The problem with the obliques (shortly mentioned already in 2.2.6) is one of the most difficult in the valency frame extraction. The obliques dependents can be both arguments and adjuncts and there is no exact rule to decide it. Sometimes even both options may be correct depending on the context (44). In the following

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	79.11	79.92	3.88	93.92	94.43	8.39
arg descr	83.36	83.97	3.67	71.42	72.6	4.13

69.4745.3687.3942.0171.646.7284.040.656.982.4921.192.29

Table 2.7: Evaluation of the coordination unit on the Czech and English data.

	CS	**EN**
# of verbs	1755	2153
x: # of *obl*s on one verb	% of verbs with x *obl*s	
1	30.0	35.4
2	7	6.7
3	0.3	0.5
4	0.0	0.1

Table 2.8: Portion of verb occurrences (computed from their overall number in the first row) with different numbers of oblique dependents.

paragraphs, the extent of the problem is discussed and then two heuristics its solution are presented.

(44) "Má kamarádka se nad řekou*[obl]* velmi podivovala."
= "My friend was very surprised at the river." (the river was surprising)
= "My friend was very surprised above the river." (she stood on a bridge)

The following tables[3] show the importance of this task because of high frequency of the oblique dependents. As we can see from Tab. 2.8, around 40 % of verb occurrences usually have an oblique dependent. Note that a verb can be the head of multiple oblique dependents. Also obliques need not to depend only on verbs: by their definition, they can be attached also to adjectives or adverbs.[4] But in most cases, they are headed by a verb, as the Tab. 2.9 shows.

The Tab. 2.10 shows the counts of obliques in comparison with other deprels being included into the valency frame. The importance of obliques follows from this statistic, too, because they appear to be in the three most frequent dependents from the set of deprels we take into account, with nominal subject and

[3] This statistic can be obtained running the `treebank_examiner.sh`

[4] *UD web* 2023, `https://universaldependencies.org/u/dep/obl.html`

	CS	**EN**
# of *obl*s	1016	1236
% of *obl*s on verb	78.0	88.7

Table 2.9: Portion of oblique dependents which depend on verbs.

direct object. Each of these three deprels is more frequent than other complements altogether. But we must be aware of the fact, that all the other deprels will be certainly included into the frame, while many of the obliques may be left out as adjuncts, so the final portion of obliques among the other frame arguments will differ.

	CS	**EN**
# of verb complements	3233	3493
1	*nsubj*: 30.4 %	*nsubj*: 31.6 %
2	*obj*: 27.9 %	*obl*: 31.4 %
3	*obl*: 24.5 %	*obj*:24.9 %
other	17.3 %	12.1 %

Table 2.10: Comparison of frequency of oblique verb dependents on verbs with other complements relevant for valency. Among "other" deprel the *iobj*, *csubj*, *ccomp*, *xcomp* and *expl* deprels are combined.

For a better idea about what are these oblique dependents like, let us have a look at their form in the sentence. In the Tab. 2.11, the most frequent forms (cases and connected adpositions) are shown. For both languages holds, that the most frequent use is with the preposition $v = in$. The table shows both the count of single occurrences of the form and the count of unique verb lemmas the oblique form is used with (which must be always smaller or at most equal as the former count).

	CS		**EN**	
# of *obl*s on a verb	1099		790	
	form	%	form	%
1	v+[Loc]	13.7	in+	13.6
2	do+[Gen]	3.2	to+	5.0
3	na+[Loc]	2.3	by+	4.0
4	[Ins]	2.3	on+	3.5
5	na+[Acc]	1.7	with+	3.2
6	k+[Dat]	1.7	as+	2.7
7	s+[Ins]	1.7	for+	2.6
8	z+[Gen]	1.5	from+	2.1
9	po+[Loc]	1.5	at+	2.1
10	[Gen]	1.2	into+	1.3

Table 2.11: Comparison of different forms of oblique dependents on verbs. For each language, ten most frequent oblique forms are shown with their percentage of the whole number of obliques on verbs (stated in the first row). The form is described by an adposition and a morphological case in brackets (where it is applicable).

When examining the significance of the oblique complements, we may have look also on the gold annotation (see 2.7.1 - 2.7.2). Tab. 2.12 shows the portion of obliques on verbs (only in the sentences od PUD processed by the gold annotation), that are arguments or adjuncts. We can see that oblique actants are more frequent in English, which will affect the results of the oblique treating unit.

gold data	**CS**	**EN**
# of *obl*s on a verb	93	119
- actants %	15.7	35.3
- adjuncts %	84.3	64.7

Table 2.12: Portions of actants and adjuncts among oblique complements of verbs, measured on gold data.

2.4.5 Obliques handling methods

There is no reliable rule deciding whether an oblique should be considered an actant or an adjunct, except of the ones that bear *obl:agent* or *obl:arg* deprels (see 1.3.4) and are included among the actants automatically (and nothing regarding the obliques in the next lines relates to these cases). In this work, several approaches are tested. Two of them are trivial: to mark all the obliques as actants or as adjuncts. Other three are use a more elaborated heuristic presented in the following paragraphs. Another option affecting the obliques is frame reduction, described in 2.4.6. All of these approaches work on the monolingual extraction level, i.e. using characteristics of the concerned frame or through comparison with other frames of the same verb. Yet one more method would be to decide the inclusion of obliques on the bilingual frame alignment level. If some of the obliques were not decided during the monolingual extraction, they might be included into the frame with an specific attribute saying, that their inclusion is not definitive, and it will be later reconsidered according to the inclusion of the counterpart in the other language. This method is not implemented in this work, because the two parts (monolingual extraction and cross-lingual) linking are strictly separated and it would involve major changes of the program to implement this approach. It is an option where the work could be possibly extended to in the future.

For the non-trivial monolingual-level approach, we need to remember the characteristics of actants (see 1.2.3. Firstly, they should be unique within the frame. E.g. a verb can have only one subject; although it can be a coordinated subject including multiple referents, it still acts as one actant towards the verb. On the other hand, in the case of adjuncts, one position can by occupied by many of them at once, as shown 2.5, where there are two specifications of place (we rely on the given parsing here and ignore the option of analysing *city* as a *nmod* depending on *river* as in "What kind of a river?" - "A river in a big city"). However, it may be hard to decide, whether two complements are of the same type or not, because they do not need to have the same form, as we can see also from the given example, where they have different prepositions. And vice versa, two distinct complements can have the same form. In (45), the verb has two objects in the bare accusative, but one of them is the addressee of the teaching

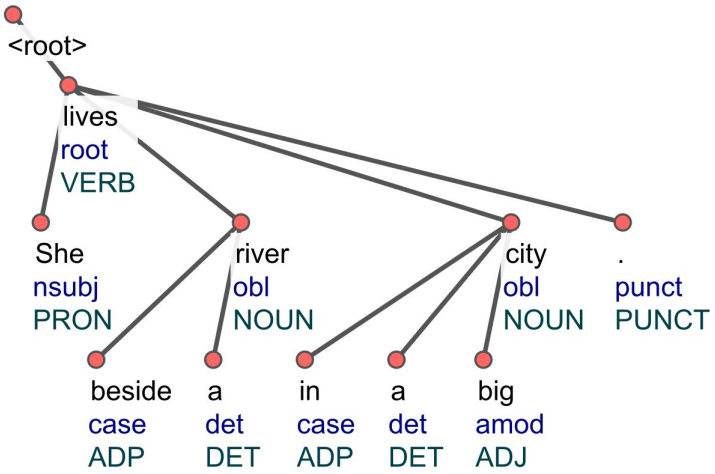


Figure 2.5: Example of a sentence with two oblique dependents acting as adjuncts describing place.

and the other one is the taught matter. In (46), there are two complements in prepositionless instrumental, where the first is an obligatory adjunct describing the route (although it not need to be necessarily in this form, we could use also a preposition *přes* = *through* with an accusative) and the other one is an optional adjunct meaning the means of transport. Because of this unclarity, I decided not to take the repetition of same form into consideration when deciding whether an oblique complement is an actant.

(45) "Učím děti$_{[Acc,iobj]}$ němčinu$_{[Acc,obj]}$." = "I teach children German."

(46) "Projíždím Prahou$_{[Ins,obl]}$ vlakem$_{[Ins,obl]}$."
= "I pass through Prague by train."

Another attribute of actants is that they are specific for the given verb and could not be used with any verb, while adjuncts can be used with almost any verb. So if there is an oblique complement its form (with a particular preposition and in a concrete case) used with a given verb lemma, we examine its co-occurrence with other verb lemmas. The criterion is following: if it occurs with a large number of other lemmas, it is rather an adjunct, if it occurs with a small number of other lemmas, it is rather an actant. We may measure the portion of the co-occurring verb lemmas of the overall verb lemmas number. But we must still remember that this rule does not hold absolutely: there can be a frequent actant form. This is of course the case of subjects and direct objects, which are fortunately labeled with their specific deprel, but it can be also true for some oblique actants. The portion number and the portion of co-occurring lemmas was shown in the

The other way around, the use of a rare form does not guarantee that the complement is an actant. It makes sense to examine how often the form occurs with the given verb lemma. Of course the actant can be optional or obligatory, but unexpressed in some context and its frequent absence could lead us to its incorrect exclusion from the frame based on its rarity. But we can suppose that there should still be many cases, where the actant is expressed and taking the frequency of its co-occurrence with the verb lemma into the account avoids incorrect considering

of rare adjuncts as actants. The rule sound as follows: if the oblique complement in the given form occurs often with the given verb lemma, it is rather the actant, if not, it is rather an adjunct. Again, we may quantify the frequency in relatively.

The combined criterion is thus this: the more frequent is the oblique form with the given verb lemma and the less frequent it is with other verb lemmas, the bigger is the chance that it is the actant of the verb lemma.

As the described heuristic uses the overall frequency of a oblique with both the same verb and with other verbs, it must be performed after the whole dictionary is finished. For each oblique complement, we count its relative frequency with its own verb and with all verbs in the dictionary (under relative frequency I mean the portion of the frame occurrences with the oblique out of all occurrences). Then we have to find out whether both relative frequencies are low or high and how to combine them together. Instead of trying multiple values that would divide the obliques into two groups in both cases, I used the comparison with the average value of the relative frequencies. According to the combined criterion, in order to mark an oblique as a frame argument, its relative frequency within its own verb should be high while within all verbs it should be low. In our case, if the former is above-average and the latter is below-average, the oblique is included into the frame as a certain argument. If both values are the other way around, it is left out as a certain adjunct. If one of the values indicates the argument, but the other would lead to the adjunct, the situation must be further examined. Three variants of the heuristic appear here. Either we mark the undecided obliques as actants or as adjuncts or we find out which of the two parts of the combined criterion is stronger, i.e. relatively further from the average. More precisely, the percentage distance of the oblique value from the average value to the extreme value is counted and which distance scores higher, its criterion wins.

In the table Tab. 2.13 the five approaches are compared: consider all obliques actants (always actants), consider them all adjuncts (always adjuncts) or to use the heuristic with the three further options: consider the undecided obliques actants (heuristic: actants), adjuncts (heuristic: adjuncts) or decide it according to the stronger part of the criterion (heuristic: middle).

The variant always choosing actants is actually identical with the deactivation of the whole unit, which can be seen from the BASE-PLUS results. Between the MINUS and FULL extractors, there is a small difference, because the obliques are taken into the frame and even if they are later removed, they affect the work of other units, namely the coordination handling unit, in the meantime. Another difference is that the basic algorithm does not include the obliques at all, even if they have *obl:arg* or *obl:agent* deprels, while the obliques handling unit keeps these oblique types in all variants, even when all other obliques are removed as adjuncts.

The results are not very good. In Czech, the unit does not improve argument selection at all, it rather worsens it. Slight improvement is observable in English, although mostly not in the combinations with other extraction units. An interesting point is the improvement in the English BASE-PLUS test even for the scenario, where all obliques are considered actants, which should be obviously worse than the opposite, "always adjunct" variant. This is true for Czech, but in English, there is larger portion of oblique actants (see Tab. 2.12). Although there are still not more frequent than the oblique adjuncts, their selection increases the

recall more than lowers the precision and the final f_1-score is higher than in the BASE variant (for details about the evaluation metrics see 2.7.6 - 2.7.9).

		CS		EN	
		BASE PLUS	MINUS FULL	BASE PLUS	MINUS FULL
always actants	arg ID	-18.33	-142.98	2.62	-19.04
	arg desc	0.0	2.07	3.59	1.93
heuristic: actants	arg ID	-10.91	-101.73	5.18	-10.41
	arg desc	0.0	2.07	3.04	1.55
heuristic: middle	arg ID	-10.15	-78.19	2.19	-7.09
	arg desc	-1.74	0.69	2.07	1.23
heuristic: adjuncts	arg ID	-2.78	-20.3	4.16	2.78
	arg desc	0.0	0.69	1.67	1.04
always adjuncts	arg ID	0.0	-1.73	0.0	-1.16
	arg desc	0.0	0.0	0.0	0.0

Table 2.13: Evaluation of oblique handling unit for the three mentioned approaches measured on Czech and English data. For simplicity, only improvements are shown here always for BASE-PLUS and MINUS-FULL tests and for argument ID and argument description evaluation values.

In this situation, I decided to use the fourth variant with heuristic preferring the adjuncts as the default setting of this unit (this can be changed in the configuration file, see E). It is the only scenario, showing improvement in both English test and at the same time the worsening on the Czech side is not so big. Moreover, it corresponds to the distribution of actants and adjuncts among oblique complements, where there are significantly more of them adjuncts, but not all. This setting of the oblique treating unit is used also in all other tests in this work. Having chosen the parameter, we can have a look to the complete results for this unit, presented in the Tab. 2.14. They show that the contribution of this unit is below the expectations flowing from the extent of examination dedicated to the topic of oblique complements.

2.4.6 Frame reduction (`frdc`)

After extracting all valency frames from a given treebank, another unit for reducing their number is applied. The idea is based on the observation, that arguments of one frame can form a subset or a superset of arguments of another frame, as

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	79.11	78.53	-2.78	95.37	94.43	-20.3
arg descr	83.36	83.36	0.0	72.41	72.6	0.69

69.4745.3687.3942.0170.7446.2787.0341.44.161.672.781.04

Table 2.14: Evaluation of the oblique handling unit with the adjunct-preferring heuristic, measured on the Czech and English data.

	CS		EN	
	PLUS	FULL	PLUS	FULL
# of frames	1194 → 1111	1177 → 1128	1455 → 1242	1333 → 1224
- 0 superframes	940	1070	920	1084
- 1 superframe	238 → 161	107 → 58	491 → 306	244 → 139
- 2 superframes	16 → 10	-	43 → 15	5 → 1
- 3 superframes	-	-	1	-

Table 2.15: Statistics of the frame reduction unit: counts of frame types having various number of superframes "before → after" the frame reduction. The counts are counted from frame types here, not individual frame occurrences as in statistic for other units (see C). The counts are given for the reduction coefficient equal to one.

in the examples (47) and (48). In such cases a question arises, whether they are truly two separate frames or they are actually identical, with the larger frame having the additional arguments optional, or vice versa, the smaller frame having some necessary arguments unexpressed, because they are understood from the context.

(47) "Otec*[nsubj,Nom]* prodal auto*[obj,Acc]*."
= "The father sold the car."
→ [nsubj : Nom] [obj : Acc]

(48) "Otec prodal auto starostovi*[obl,Dat]*."
= "The father sold the car to the mayor."
→ [nsubj : Nom] [obj : Acc] [obl : Dat]

(49) "Otec prodal auto za*[case]* velké peníze*[obl,Acc]*."
= "The father sold the car for big money."
→ [nsubj : Nom] [obj : Acc] [obl : za+Acc]

(50) "Otec prodal auto starostovi za velké peníze."
= "The father sold the car to the mayor for big money."
→ [nsubj : Nom] [obj : Acc] [obl : Dat] [obl : za+Acc]

All frames belonging to one verb lemma can be organized into a structure, where each frame is connected with its direct subframes, i.e. frames including all if its arguments, but also some additional ones. This structure is actually not a tree but a DAG (directed acyclic graph), because one frame can be a direct subframe for multiple superframes, as the frame in (50), which is a direct subframe two frames in (48) and (49). However, this does not happen very often: there are only few cases of frames having two or three superframes in PUD treebanks, and no frames have more superframes (see Tab. 2.15).

Although it is possible that a superframe is actually identical with its subframe, only with argument elided, the elision happens in most cases for subjects, which are added to every frame. Thus the frame reduction takes place only in the opposite direction, supposing that some arguments of a subframe are actually optional and the frame should be merged with its superframe omitting this

argument. This direction if the reduction is also simpler to implement, because (almost) each frame has only one superframe to be merged with, while the other way around, there might be multiple subframes as candidates for merging. In order to perform this type of reduction, we need to transform the DAG into a tree (on larger data there might be many frames with multiple superframes). To do it, a simple technique is used: the superframe with greatest *subtree number* is chosen. For each frame we define a *subtree number* as the sum of occurrences of the frame itself and occurrences of frames in its subtree, i.e. all its (direct or indirect) subframes. The tree structure of the frames is used not only for the frame reduction, but also for the dictionary visualization as one of the options (see D).

The reduction itself proceeds as follows: For each frame we compare the number of its occurrences with the subtree numbers of all its subframes. If the frame occurrences number is greater than the subtree number multiplied by a reduction coefficient (a unit parameter), the whole subtree is merged with the frame concerned, omitting all arguments absent in this frame. The reason we merge the whole subtree is that if the direct subframe meets the condition, each frame of the subtree also does, because its subtree number is even smaller. The subframes may be iterated multiple times, because merging with one subframe increases the number of occurrences of the frame concerned, which is used in the comparison, so after one pass through the subframes there may be some newly appropriate for merging, that were not before. When there is no subframe fulfilling the reduction condition, the reduction is called recursively on the subframes left.

The whole frames reduction process is parameterized by the already mentioned reduction coefficient. Several its values were tested in order to find the optimum, particularly 0, 1, 2, 3 and 4. If the reduction coefficient is equal to zero, the reduction always happens, because any number of frame occurrences is greater than zero times the subtree number, all frames having a superframe are merged with it. On the other side, if the reduction coefficient goes to infinity, the reduction never takes place, so the unit can no effect and the improvement should go to zero.

The overview of frame reduction results depending on the reduction coefficient is shown in Tab. 2.16. Although I believe the idea behind the frame reduction is correct, this heuristic does not improve the results at all according to the results, regardless on the value of the reduction coefficient. Therefore I decided not to include this unit by default to the extraction process. The tests of all units (of the main extractor and of the language-specific extractors) are performed without the frame reduction. However, I keep the code of the unit and the option to activate it and to set the reduction coefficient in the configuration file (see E.

2.5 Czech and Slovak extension

Czech and Slovak are closely related languages and share many grammatical features. Their language specific extractors are almost identical, they differ only in the lemma of the verb *to be* and in the list of modal verbs. Their extractors are actually realised as one Czech-Slovak frame extractor with inherited classes for Czech and Slovak adding the appropriate lemma forms. However, there is no Slovak PUD treebank, so the frequency measurements and evaluations are performed

reduc coef		CS		EN	
		BASE PLUS	MINUS FULL	BASE PLUS	MINUS FULL
0	arg ID	-56.49	-56.91	-97.77	-48.77
	arg desc	-42.19	-4.12	-34.22	-4.69
1	arg ID	-11.58	-8.62	-23.65	-16.73
	arg desc	-7.93	-0.69	-7.38	-1.57
2	arg ID	-4.6	-0.72	-7.89	-8.09
	arg desc	-4.57	-1.02	-2.89	-0.59
3	arg ID	-3.21	0.9	-2.72	-4.52
	arg desc	-1.14	0.0	-0.82	-0.4
4	arg ID	-3.21	0.9	-1.47	-1.19
	arg desc	-1.14	0.0	-0.2	0.0

Table 2.16: Evaluation of frame reduction unit for various values of reduction coefficient, measured on Czech and English data. For simplicity, only improvements are shown here always for BASE-PLUS and MINUS-FULL tests and for argument ID and argument description evaluation values.

only for Czech. The frame-level units of the Czech-Slovak extractor are verbal adjectives 2.5.1 and exclusion of modal verbs 2.5.2, while its argument-level units are passive voice 2.5.3, adding other missing cases 2.5.4, adding missing nominatives to subjects 2.5.5, changing participles to finite forms 2.5.6 and numerative 2.5.7.

2.5.1 Verbal adjectives (`vadj`)

In the general extractor, the frames are based only on words bearing a *VERB* upostag. The Czech-Slovak extractor explicitly includes also verbal adjectives, that are considered adjectives by their upostag (*ADJ*), but have also a *VerbForm* feature marking them as participles (*Part*). The verbal adjectives may be active or passive and at the same time imperfect (describing actually happening action) or perfect (describing past, already finished action). Imperfect active verbal adjectives are e.g. *dělající* (=*doing/making*) or *beroucí* (==*taking)*, their perfect counterparts (rarer) are *udělavší* (=*having done/made*) or *sebravší* (=*having taken*). Examples of perfect passive participles are *udělaný* (=*done/made*) or *zakrytý* (*covered*), with imperfect counterparts *dělaný* (=*being made*) and *krytý* (=*being covered*). Actually, only short forms of passive verbal adjectives used exclusively in nominal predicates (*dělán*, *zakryt* etc...) have the *VerbForm=Part* feature, which complicates the work of this unit (for the solution see 2.7.5).

Verbal adjectives, especially the passive ones, are quite frequent, so by ignoring them we would lose a lot of information. They preserve the valency of the original verb, with the exception of the subject, which is occupied by the word heading them (and passive participles must undergo depassivization, of course - see 2.5.3). Moreover, in English and many other languages, the corresponding forms are annotated as verbs (being considered inflected, not derived), so their inclusion in Czech and Slovak is desirable also because of better bilingual linking.

	verbal adjectives	as attributes
total	267	63
- active	62	60
- passive	205	3

Table 2.17: Statistics for the passive voice unit: Counts of various argument substitutions for both types of passive.

On the other hand, in Czech and Slovak tradition, verbal adjectives are considered adjectives derived from verbs, rather then inflected forms of the verbs, so the lemma of the verbal adjectives is still an adjective, not the original verbal lemma. This causes that there are often two separate records in the dictionary (one for the verb and another for the verbal adjective). In manually annotated Czech UD treebanks, the information about the original verb is marked in the last field (*MISC*) of the CoNLL-U format, but the UDPipe does not perform such annotation, so in the automatically annotated treebanks (which we aim to use for the purpose of valency frame extraction) the verb lemma would be missing anyway. Therefore I decided not to take this additional information into account. In the end, the duplicate records in the dictionary are not such a big issue (the loss of the information caused by ignoring the verbal adjectives would be worse) and the two records are often close to each other in the alphabetical ordering of the records, so it does not harder the dictionary viewing as much. Another problem is that in PUD, the annotation of passive verbal adjectives is often incomplete and they lack the *VerbForm* feature. This problem is compensated by an adjusted version of gold frames (see 2.7.5).

According to 2.17, there are 267 verbal adjectives in the Czech data, which makes 13.2 % of 2022 frame instances; most of them are passive. Apart from their inclusion among the extracted frames, also their subject may be extracted, if they are dependent on a node as their attributes (51). For passive verbal adjectives, *nsubj:pass* is assigned to the subject (and later may be transformed to object by the depassivization unit, if activated). But most of the verbal adjectives are actually not in such attributive position, but rather they are predicates, often forming the proper passive voice (52). In this case, the parent note is not taken as the frame subject, because the subject is among the verb's children or it is unexpressed. However, most of the cases, where the verbal adjective is in the attributive position, are active, which leads us to an interesting observation, that the passive verbal adjectives are more often in predicates, whereas the active ones are rather attributes.

(51) "zajíc **skrytý** v poli" = "a rabbit **hidden** in the field"

(52) "Zajíc je **skrytý** v poli." ="The rabbit is **hidden** in the field."

In the case of this unit, the result strongly depend on the golden data set used. If the lemmas of verbal cores of the gold frames would be adjectives in accordance with the actual situation of PUD, the result for lemmas of the tested extractors would be also 100 % (see results measured on the gold data 2.7.10). Here, the data with appropriate infinitives is used, so the tested extractors score

	BASE	PLUS	impr	MINUS	FULL	impr
verb ID	91.24	98.8	86.3	91.24	98.8	86.3
lemmas	100.0	100.0	0.0	100.0	100.0	0.0
arg ID	94.43	93.03	-25.13	96.2	96.79	15.53
arg descr	72.6	67.15	-19.89	97.65	97.21	-18.72

Table 2.18: Evaluation of verbal adjectives unit.

less than 100 % (obtaining infinite worsening). However, the important part is significant improvement of frame selection, even for the price of worsening the argument extraction. In the PLUS extractor, the depassivization unit (see 2.5.3) is not applied, so the newly extracted passive verbal adjectives have incorrect arguments, which is the main cause of the worsening.

2.5.2 Modal verbs exclusion (`mdex`)

Unlike in English, modal verbs in Czech and Slovak are labeled with *VERB* upostag, so they are included into the dictionary by default. In order to avoid including them, if desired, the Czech and Slovak extractors have lists of modal verbs and they compare lemma of every verb node to the lemmas in this list. In the case of agreement, the verb is not included into the dictionary. Another problem is that the full-meaning verb infinitive is missing the subjects, which is attached as the child of the modal verb, i.e. the sibling of the infinitive. The situation is shown in the Fig. 2.6 for the sentence "Můj bratr dnes musí psát domácí úkol." ("My brother must write the homework today", but the tree would be different in English – the modal verb would be marked as AUX and headed by the full-meaning verb). The subject is added to the infinitive automatically because of the corresponding main unit (2.4.1).

The lists of modals verbs for Czech and Slovak extractor slightly differ not only in the actual infinitive form, but also in the general choice of the verbs, because the languages have different traditional perception of what is considered a modal verb. Common and etymologically related modal verbs in both languages are (in Czech/Slovak order): *muset/musieť* (=*must, have to*), *moci/môcť* (=*can, be able to*), *mít/mať* (=*should, be supposed to*), *smět/smieť* (=*may, be allowed to*), *chtít/chcieť* (=*want to*) and *hodlat/hodlať* (=*intend to*). The Czech has additional modal verbs *umět*, *dovést*, while the Slovak has a modal verb *vedieť* (they all meaning *can, know how to*, but are unrelated). Some of these verbs have also a full-meaning use. Finally, there may be also various different lists found.

The unit excluding modal verbs is by default deactivated (so it is not included in the tests of other units) but it can be activated in the configuration file (see E). The numbers of modal verb frames in the Czech PUD treebank can be seen on 2.19. The given total number 118 represents approximately 7 % of all verb occurrences in the data, which is similar as in English (see 2.6.2). The activation of this unit and thus exclusion of the modal verbs leads to worsening of the results (see 2.20), because the modal verbs are included into the gold data for Czech. The unit enables to exclude modal verbs if it is necessary, e.g. for comparison with other languages.

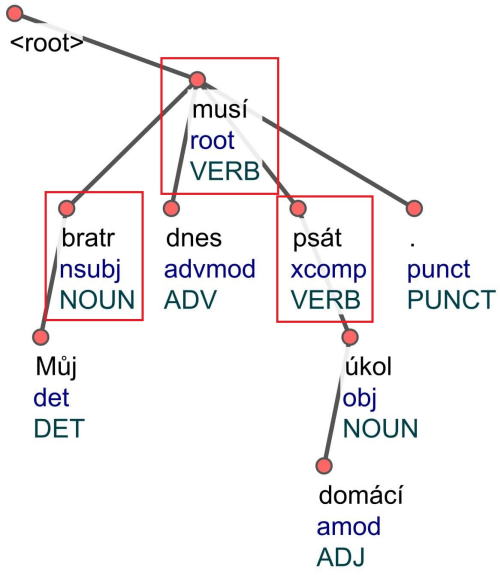


Figure 2.6: The sentence from the example with the three discussed nodes (the modal verb, its subject and the infinitive of the full-meaning verb) and their relevant attributes highlighted.

total	118		
chtít	11	muset	20
mít	23	smět	4
moci	58	umět	2
hodlat	0	dovést	0

Table 2.19: Statistics of the modal verb exclusion unit: numbers of modal verbs in the treebank.

	BASE	PLUS	impr	MINUS	FULL	impr
verb ID	91.24	89.24	-22.83	98.8	97.07	-144.17
lemmas	100.0	100.0	0.0	100.0	100.0	0.0
arg ID	94.43	95.04	10.95	96.79	97.4	19.0
arg descr	72.6	71.91	-2.52	97.21	97.24	1.08

Table 2.20: Evaluation of modal exclusion unit.

2.5.3 Passive voice (pass)

The passive voice in Czech (and Slovak) can have several forms. There is a "proper" passive expressed by the passive participle with the auxiliary verb *být/byť = to be* (53). It can express the agent of the action by the instrumental case (54). However, there are also forms of "improper" passive, which use an active verb form. They use either a reflexive construction with the patient still being a subject as in the proper way (55) or an impersonal construction, where the patient is an object, the verb is in the third person plural and the subject is unexpressed (56). The latter cannot be actually called a passive voice as the patient is the object, which is a feature of active forms, I mention it here because it allows not to mention the agent, which is a property shared with the two previous passive options. Moreover, the impersonal construction preserves the proper valency frame of the verb and does not need any modification, so only the two truly passive constructions are discussed further.

(53) "Oběd je dělán*[Pass,Part]*." = "The lunch is being made."

(54) "Oběd je dělán bratrem*[Ins,obl:arg]*."
= "The lunch is being made by the brother."

(55) "Oběd se*[expl:pass]* dělá." = lit. "The lunch is making itself."

(56) "Dělají oběd." = "(They) are making lunch."

The passive occurrences of a verb usually precisely correspond by their valency frames with the active ones, they have just undergone the passive transformation. When extracting a valency dictionary, it is useful to perform their depassivization, i.e. transform all passive constructions into active in order to eliminate redundancy, otherwise both active and passive usages of the verb would result into two different valency frames.

The proper passive is detected if the auxiliary verb (with *AUX* upostag) *být* headed by a verb in passive participle form (*VerbForm=Part* and *Voice=Pass* features) is found. The subject in the nominative is then replaced by the object in accusative. If there is an oblique complement in instrumental, it is transformed only if there is an *obl:arg* deprel specification (which should be present in these cases in Czech treebanks), as in (54), otherwise it could happen, that a true adjunct in instrumental would be considered an actant, as in (57), where the instrumental does not express the agent, but rather the environment, where a motion action takes place.

(57) "Dřevo je neseno městem*[Ins,obl]*."
= "The wood is being carried through the town."

The proper passive construction (53) can be said also reflexively (55) without any change in the meaning. However, there is a syntactic difference, that the reflexive passive cannot express the agent using the instrumental case, as the proper passive optionally can (54). In the cases of the reflexive passive, the verb is in the active voice, the patient in nominative is its subject and the reflexive pronoun in accusative is its formal object. This corresponds to an active frame, where the patient is an object in nominative and the subject is unexpressed (the

	proper	reflexive	
	FULL	PLUS	FULL
total	205	49	
nsubj → obj	199	34	41
csubj → ccomp	5	1	1
obl → nsubj	32	1	1

Table 2.21: Statistics for the depassivization unit: counts of various argument substitutions for both types of passive.

reflexive pronouns does not correspond to anything in the active counterpart). In order to extract the right valency frame, we need to perform such transformation. But it cannot be done automatically in all cases, unlike the occurrences of proper passive, because there is a possibility that the sentence is properly reflexive, i.e. the subject is both the agent and the patient, as in (58). These cases cannot be distinguished without any additional information. Fortunately, UD offers a way to indicate, whether the reflexive pronoun is a proper object of the verb, meaning that the patient is identical with the agent, or if it is a part of the periphrastic passive voice. In the former case, the reflexive pronouns should be marked with an *obj* deprel as any other (pronominal or nominal) objects, but they are often incorrectly annotated with a *expl:pv* deprel, reserved for inherently reflexive verbs (59). However, this does not prevents the correct recognition of the reflexive passive, because in that case, the reflexive pronouns are marked with *expl:pass* deprel (55) (see also description of *expl* deprel in 1.3.7). This distinction allows to perform the transformation correctly.

(58) "Otec se*[obj]* sprchuje." = "The father is showering himself."

(59) "Otec se*[expl:pv]* bojí." = "The father is afraid."

(60) "Bylo zjištěno, že stroje přestaly*[csubj:pass]* fungovat." (proper passive)
∼ "Zjistilo se, že stroje přestaly*[csubj:pass]* fungovat." (reflexive passive)
= "It was found that the machines had stopped working."

In both types of passives, the proper and the reflexive one, the depassivized argument can be either nominal or clausal. In the former case, the nominal subject *nsubj* is replaced with a nominal object *obj*, in the latter case, the clausal subject *csubj* is changed to clausal complement *ccomp*. Clausal cases of both passives are shown in the sentences (60). Frequency of all phenomena discussed is summed up in Tab. 2.21. The proper passive is treated only in the FULL mode, because the participles used in it have *ADJ* upostag and are included into valency dictionary only when verb adjective unit is activated (see 2.5.1). For one passive predicate, multiple argument transformations may take place, so the counts for individual transformations does not need to sum up to the total count of passive clauses). There is typically a subject (nominal or clausal) transformed to the object, although it does not need to be the necessarily, and possibly also a oblique agent, transformed into the subject. The clear contribution of the depassivization can be seen in the Tab. 2.22.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	94.43	95.69	22.62	93.22	96.79	52.65
arg descr	72.6	73.68	3.94	87.26	97.21	78.1

Table 2.22: Evaluation of the depassivization unit.

2.5.4 Adding missing cases (`case`)

In Czech and Slovak, nominals must always take an appropriate case according to their function in the sentence. The proper case can be usually derived from the word form, its ending in particular. However, sometimes it is not clear in which case the word is. Most of these word are foreign words and proper names that cannot be declined in the languages' declension system as usual. They lack the *Case* feature in their annotation. After the extraction, they would lead to creation of a separate frame with a caseless argument, different from a corresponding frame with the corresponding argument including the correct case, extracted from another sentence, where the argument was filled by a usual word instead of a foreign proper name. To prevent this mistake, addition of cases takes places for all nominal arguments.

In the case of subjects, the nominative case is automatically added to the argument (see the next unit). The case of other arguments cannot be derived from their deprel so reliably, although clear majority of objects take the accusative case, but the situations when they do not are the interesting part of the output, so it would harm to add the accusative case to all of the unclear occurrences. The addition of cases to other arguments takes place at the end of the extraction by comparison with other frames of the verb. A simple heuristics is used here: we find a corresponding frame with all arguments identical, but with a filled cases in all places, where they are missing in the original frame. If there are multiple such frames, the one with most occurrences is chosen. The corresponding cases are added to the argument, where they are missing and the frames concerned are merged.

The frequency of the cases added is shown in the Tab. 2.23. The most added case in the nominative, because it is added to all unexpressed subjects added by the main, subject-adding unit, which does not specify any argument form (see 2.4.1. In the FULL variant, these cases are already having been treated by the separate unit (see 2.5.5), so the count of the nominative is remarkably lower. The unit itself affects only argument description, but that has influences the main extractor's oblique dealing unit, so the results (see Tab. 2.24) show small improvement also in the argument IDs. Nevertheless, the main contribution of the case adding is to the argument description.

2.5.5 Adding nominatives to subjects (`nomi`)

Adding elided subjects is a part of the general extraction algorithm (see 2.4.1), but further description of these arguments is left on the individual language-specific extractors. The Czech-Slovak extractor only adds that the subjects are in nominative case. Adding nominative is applied also on all present, non-elided subjects, that from some reason (e.g. because of foreign origin) lack the case

	PLUS	FULL
# of arguments	3685	4257
total	95 (2.6 %)	12 (0.3 %)
- nominative	82	1
- genitive	3	0
- dative	1	2
- accusative	6	7
- locative	2	2
- instrumental	0	0

Table 2.23: Statistics for the case adding unit: counts of individual cases added to the arguments.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	94.43	94.65	3.95	96.6	96.79	5.59
arg descr	72.6	76.74	15.11	96.89	97.21	10.29

Table 2.24: Evaluation of the case adding unit.

feature. Because elided subjects are quite frequent phenomenon, the improvement of argument description reached by the unit is significant, as we can see from Tab. 2.26. The minor changes in argument ID are not directly caused by the unit, but rather by other units where the argument form is important for argument selection (if we run the test with all units deactivated, the argument ID results would be identical for all both reference and tested extractor). If the unit is deactivated, many of the formless subjects get the nominative form thanks to the case-adding unit (see 2.5.4), but this unit works only when there are other occurrences of the frame, where the subject actaully has the nominative form. The separate nominative unit works automatically for all subjects, regardless of the other occurrences. The high frequency of the additions is visible in the Tab. 2.25. The evaluation in Tab. 2.26 show that the unit affects mainly results of the argument description, but in combination with combination with the depassivization unit 2.5.3 it improves also the choice of the right arguments, because the depassivization occurs only for nominative subjects.

	PLUS	FULL
# of arguments	3680	4257
total	648 (17.6 %)	906 (21.3 %)

Table 2.25: Statistics for the nominative adding unit: counts of originally formless subjects that were assigned the nominative case.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	94.43	94.43	0.0	96.29	96.79	13.48
arg descr	72.6	93.5	76.28	77.79	97.21	87.44

Table 2.26: Evaluation of the nominative adding unit.

	PLUS	FULL
# of arguments	3680	4257
total	56 (1.5 %)	57 (1.3 %)

Table 2.27: Statistics for the participle form unit: counts participles whose form was changed to finite.

2.5.6 Changing participles to finite forms in clausal arguments (`pfin`)

Czech and Slovak clausal arguments may obtain either infinitives (61) or finite verbal forms (usually with a conjunction). If the verb is in present tense, indicative mood and active voice, it is itself used in a finite form, without the auxiliary (62). Participles occur in compound verbal forms, mostly accompanied by the auxiliary verb *být*/*byť* (*to be*) in a finite form. They can be either past (also called active), forming past tense (63) and conditional mood (64), or passive, used to form passive voice (65). It is clear that the form imposed on the clausal argument by the frame verb is finite, so it is necessary substitute the *Part* feature of the main verb with the *Fin* form of the auxiliary. The preference of the finite verb form is already implemented in the general algorithm 2.4.2. But in the third person of the past tense, the auxiliary verb is omitted and the participles are used on their own (66). In that case, explicit substitution of the participial forms of the clausal arguments with finite forms takes place. In the Czech PUD, it occurs 51 times, which constitutes 1.2 % of all frame argument occurrences. The frequency of the described replacements is not very high (see Tab. 2.27). They not effect the choice of arguments, even in combination with other units. The improvement of argument description is positive, but small (Tab. 2.28).

(61) "Snažíme se běhat." = "We are trying to run."

(62) "Říkám, že vařím." = "I say that I will cook."

(63) "Říkám, že jsem vařil." = "I say that I cooked."

(64) "Říkám, že bych vařil." = "I say that I would cook."

(65) "Říkám, že jsem vařen." = "I say that I am being cooked."

(66) "Říká, že vařil." = "He says that he cooked."

2.5.7 Numerative (`numr`)

Similarly as in other Slavic languages, Czech and Slovak nouns take the genitive case after some numerals. Numerals two, three and four cause the noun in plural

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	94.43	94.43	0.0	96.79	96.79	0.0
arg descr	72.6	72.93	1.2	96.84	97.21	11.71

Table 2.28: Evaluation of the participle form unit.

	PLUS	FULL
# of arguments	3684	4257
total	66 (1.8 %)	73 (1.7 %)
- nominative	18 (1.5 %)	23 (1.3 %)
- accusative	48 (1.5 %)	50 (1.3 %)

Table 2.29: Statistics for numerative unit: frequency of originally genitive arguments, that were marked as numerative and later transformed to nominative or accusative.

following them to take the case corresponding the role of the phrase in the sentence, as in English: *dva[Nom] psi[Nom]* = *two dogs*. However, after the numerals five and higher, the nouns take the genitive case instead of the nominative, accusative and vocative, while the numeral preserves its original case (*pro pět[Acc] psů[Gen]* = *for five dogs*). This numerically caused genitive form is called *numerative*. Note that in other cases than the mentioned ones, the original case is still preserved by the whole phrase (*o pěti[Loc] psech[Loc]* = *about five dogs*). The same rules as for the high numbers are applied after several other non-number numeral expressions (*kolik* = *how many, how much*; *několik* = *several*) and for any noun describing an amount (*hrst[Nom] zrn[Gen]* = *handful of grains*), which is already similar as in English (the Czech genitive often corresponds to the English preposition *of*) and it is actually the way, how the numerative historically evolved (that the numerals concerned began to be perceived as an amount-describing noun).

According to the UD principle of the primacy of content words (see 1.1.8), the numerals are syntactically dependent on the noun, so in the cases of numerative, the case of the valency argument would be incorrectly extracted as genitive. To prevent this, it is necessary to check the children of the argument in genitive, whether some of them is not a numerative-causing numeral, which is in the Czech data marked by a specific relation (*nummod:gov* for numbers and *det:numgov* for the other expressions). If yes, the case of the numeral child is taken into the frame as the proper argument case (i.e. either nominative or accusative; nouns in vocative should not be considered valency arguments). Counts of numeratives transformed into nominatives and accusatives are shown in the Tab. 2.29. It shows up that the numerative unit is helpful rather in combination with other units Tab. 2.30.

2.6 English extension

On the frame level, the English extractor contains exclusion of verbal prepositions 2.6.1 and inclusion of modal verbs 2.6.2, while its argument level units are adding

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	94.43	94.43	0.0	96.69	96.79	3.02
arg descr	72.6	73.26	2.41	96.4	97.21	22.5

Table 2.30: Evaluation of the numerative unit.

	BASE	PLUS	impr	MINUS	FULL	impr
verb ID	98.66	99.33	50.0	98.66	99.33	50.0
lemmas	100.0	100.0	0.0	100.0	100.0	0.0
arg ID	87.39	87.39	0.0	93.48	93.48	0.0
arg descr	42.01	42.01	0.0	96.08	96.08	0.0

Table 2.31: Evaluation of the case verbs exclusion unit.

missing cases 2.6.3, adding heads of participles and gerunds 2.6.4, passive voice 2.6.5, facultative that 2.6.6 and verbal particles 2.6.7.

2.6.1 Verbal prepositions (`cvex` - case verbs exclusion)

There are frequent cases, when a verb serves as preposition. They are marked as verbs by their upostag, but bear the *case* deprel. They do not not preserve the original verb frame and work actually like common prepositions. In other languages (like Czech), they are also annotated with the *ADP* upostag. Those are reasons why I decided to exclude them from the valency frames extraction. There are 22 such cases in the English PUD treebank (1.0 % of 2131 frames): according (11 times), including (6 times), following (4 times) and compared (to) (1 time). The improvement done by this extraction unit is present naturally only in the selection of frame verbs (see Tab. 2.31).

2.6.2 Modal verbs inclusion (`mdin`)

English has a traditional definition of modal verbs distinguishing them from other verbs that can take an infinitive. According to the custom, English modal verbs are followed by a bare infinitive without the word *to*. If the infinitive is attached to the finite verb with *to*, the verb is not considered modal. According to this rule, English modal verbs are *can*, *may*, *must*, *shall* and *will* and their past forms. There are also other common characteristics of these verbs: they can form questions (67) or negation (68) on their own without a need of the auxiliary verb *to do* or they do not take *s*-ending in the third person singular (69).

(67) “Can I sing?” vs. “Do I begin to sing?”

(68) “I must not sing.” vs. “I do not begin to sing.”

(69) “He may” vs. “He begin**s**”

This approach is problematic for several reasons. Firstly, the forms *shall* and *will* are used to form the future tense and the past form *would* to form the past future

total	160		
can	34	could	20
may	10	might	5
shall	0	should	5
must	4	-	-
will	42	would	40

Table 2.32: Statistics of the modal verb inclusion unit: numbers and portions of all and individual modal verbs in the treebank.

	BASE	PLUS	impr	MINUS	FULL	impr
verb ID	98.66	95.26	-253.73	99.33	95.88	-514.93
lemmas	100.0	100.0	0.0	100.0	100.0	0.0
arg ID	87.39	87.39	0.0	93.48	93.48	0.0
arg descr	42.01	42.01	0.0	96.08	96.08	0.0

Table 2.33: Evaluation of the modal verbs inclusion unit.

tense (also called conditional mood), working a very similar way as the auxiliary verbs. Secondly, the suppletive past forms of *must*, *had to*, is not modal which is inconsistent with other modal verbs, whose past forms *could*, *might*, *should* and *would* are still modal. Moreover, the mentioned definition of modals leaves out the verb *want*, which is in many other languages (as Czech and German) clearly considered a modal verb. Lastly, there are other verbs that are usually used with *to* but sometimes can be used without it, like *dare*.

Despite all these problems, UD accepts the traditional English definition of modal verbs. But unlike in Czech, they are annotated with *AUX* upostag, being considered closer to auxiliary verbs and thus they are not automatically included into the valency dictionary. The English extraction unit for modal verbs is inactive by default, so the modal verbs are not extracted, but it can be activated in the configuration file E to include them explicitly. It lists the mentioned modal verbs (including their past forms) for being taken into the valency dictionary despite of the *AUX* upostag. This happens 160 times for the English PUD treebank, which represents around 7 % of verb occurrences, which is a similar portion as in the Czech data (see 2.5.2). The statistic for individual modal verbs is shown in the Tab. 2.32. Because the modal verbs are frequent and the English golden data are annotated according to the UD decision to exclude English modal verbs, their explicit inclusion leads to dramatic worsening of frame verb selection (Tab. 2.33). If the verb IDs of the frames do not agree, other values cannot be measured.

Unlike, its Czech-Slovak counterpart, where it sufficed to exclude verbs fulfilling the condition for modal verbs, it is not enough to include them in the same way. In addition to it, it is necessary ensure extraction of its arguments: the subject and the full-meaning verb as a clausal complement. Originally in the sentences tree, the full-meaning verb is the parent tree of the modal verb, not its argument, and the subject is attached to it.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	87.39	87.39	0.0	93.62	93.48	-2.19
arg descr	42.01	84.2	72.52	53.73	96.08	91.53

Table 2.34: Evaluation of the case adding unit.

2.6.3 Adding cases (`case`)

Except of personal pronouns, English does not uses cases. Pronouns in the position of the subject take the nominative form (*I*, *he*, *she*, *we*, *they*), in the positions of direct objects or after prepositions, they take the accusative form (*me*, *him*, *her*, *us*, *them*). Unlike the pronouns, nouns are not marked with a case feature (and the pronoun *you*, too, because it has the same in both positions). As it is a constitutive part of the argument, the frames including arguments with marked cases are considered different from the frames with caseless arguments. Sometimes there can be actually four frames (Nom-Acc, Nom-0, 0-Acc, 0-0) instead of one, if the the treebank include occurrences of both pronouns and nouns in both positions. In order to remove this redundancy, it is necessary to unify form of the arguments. This can be done either by deleting the cases or on the contrary by adding them according to the deprel. For easier comparison with other languages and also led by the effort not to throw away useful information, I decided to keep the case features and to add them to nouns, although both options should be equivalent The addition is base on the argument's deprel: *nsubj* arguments take the nominative, *obj* and *obl* arguments take the accusative. Each of the two cases covers approximately half of the arguments in both PLUS and FULL extraction. As the nouns in the golden data already have their cases, this unit gains significant improvement in argument description (Tab. 2.34).

2.6.4 Heads of participles and gerunds (`ptgr`)

Participles and gerunds can play various roles in English, i.e. take various deprels. One of them are adjectival modifiers (*amod*) or heads of adnominal clauses (*acl*). In these cases they are headed by a noun, would normally occupy their subject position. Other arguments are dependent on the verb form as usual. Moreover, the participles have passive function in this case, so the heading noun is actually an object, not a subject of the depassivized verb frame. Because the subject/object represented by the heading nouns is missing among the children of the verb node, the English extractor explicitly implements a unit adding this parent node among the frame arguments. It is added always as subject, the transformation to object in the case of the participles is done later in the separate unit dealing with the passive voice (but in this case *nsubj:pass* deprel is used instead of *nsubj*). The missing subjects would be added also by the main extractor unit (see 2.4.1), but in that case, they would be considered elided arguments and would not be identified with the actually present parent node. This English unit is performed before the main unit, so by the time elided subjects are being added, the participles and gerunds already have their parent node among their arguments. Participles in the discussed position appear 123 times in the treebank, gerunds 100 times (see Tab. 2.35). This unit reaches better results in combination with other extracting

		PLUS	FULL
# of arguments		3735	4110
participles	123	2.7 %	2.4 %
gerunds	100	3.3 %	3.0 %

Table 2.35: Statistics for the unit for participles and gerunds: counts of participles and gerunds whose heads were transformed into their arguments.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	87.39	88.82	11.34	90.25	93.48	32.92
arg descr	42.01	43.14	1.95	93.25	96.08	41.93

Table 2.36: Evaluation of the unit for participles and gerunds.

units (see Tab. 2.36).

2.6.5 Passive voice (`pass`)

A straightforward language-specific modification for English is the transformation of passive voice to active. It takes place always when the auxiliary verb *be* occurs with an *aux:pass* deprel to a head verb in participle form (*VerbForm=Part*). In such case, its subject (*nsubj:pass*) is transformed into an object (*obj*) and if there is an oblique complement preceded with a preposition *by* expressing the agent of the action, it is replaced by a nominative subject (70). Cases with a clausal subject are treated analogically, where it is transformed into clausal complement (71). Counts of all substitutions are presented in the Tab. 2.37. According to the Tab. 2.38, the unit improves the extractor results in both the selection of arguments and their appropriate description.

(70) "The dinner*[nsubj:pass]* is*[aux:pass]* cooked by a famous cook*[obl]*."

(71) "It was*[aux:pass]* found that the machines had stopped*[csubj:pass]* working."

2.6.6 Subordinate conjunction *that* (`that`)

Another part of the English extractor treats subordinate clauses that may be introduced by the conjunction *that*. This conjunction is not necessary and the

	PLUS	FULL
total frames	2153	2131
nsubj → obj	269	389
csubj → ccomp	4	4
obl → nsubj	46	38

Table 2.37: Statistics for the depassivization unit: counts of various argument substitutions.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	87.39	90.02	20.86	89.24	93.48	39.41
arg descr	42.01	46.95	8.52	81.0	96.08	79.37

Table 2.38: Evaluation of the depassivization unit.

	PLUS	FULL
total frames	2153	2131
that present	44	44
that absent	61	64

Table 2.39: Statistics for the "that" unit: counts of the cases, where the conjunction was present (and put in brackets) and where it was missing (and added in brackets).

clauses often occur without it (72). That would lead to creation of two distinct frames: one with the conjunction *that* an other one without it. Therefore this unit aims on creating only one frame with the word *that* in brackets. In cases of clauses already introduced by *that*, this conjunction is enclosed in brackets (44 cases), in the cases of finite clausal arguments without a conjunction, the word *that* in brackets is added (64 cases). However, it does not influence the argument selection and the improvement of the argument description is relatively small (see Tab. 2.40).

(72) “I say you are late.” ∼ “I say that you are late.”

2.6.7 Verbal particles (`cprt` = with *compund:prt* deprel)

The last and smallest unit of the English extractor implements inclusion of the phrasal verb particles into the verb frame. They are rather a part of the verb than an argument in a proper sense (see also 2.1.8), but this is treated later, on the level of visualization of the dictionary. The particles are denoted with *compound:prt* deprel, which occurs 70 times in the PUD treebank (1.8 % of 3805 arguments in PLUS extraction and 1.7 % of 4110 arguments in FULL extraction). The contribution of this unit seems to be marginal according to the Tab. 2.41.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	87.39	87.39	0.0	93.48	93.48	0.0
arg descr	42.01	42.65	1.1	95.36	96.08	15.52

Table 2.40: Evaluation of the "that" unit.

	BASE	PLUS	impr	MINUS	FULL	impr
arg ID	87.39	87.81	3.33	93.08	93.48	5.78
arg descr	42.01	42.53	0.9	96.08	96.08	0.0

Table 2.41: Evaluation of the verbal particles unit.

2.7 Frame extraction evaluation

The program described above in this chapter produces a monolingual valency dictionary. To ensure that the dictionary is fitting and contains correct valency frames of the language, we have to perform evaluation using appropriately designed experiments. In order to say, whether correct valency frames were extracted from a given sentence, we need to compare the results with control data that contain the valency frames, that we were suppose to extract. In this section, I characterize the data are and describe the form and process of their golden annotation (2.7.2 - 2.7.3). Most of the cases (verbs and their arguments) the program deals with are actually trivial to extract, because the treebank annotation already contains enough of important information, while handling the minority of difficult situations are the true task of this work. The evaluation process and metrics presented in this section (2.7.6 - 2.7.9) should be designed in such way so that they capture the success of the extractors mainly in these cases. Finally, the results of the overall results of the extraction process are shown and discussed (2.7.11 - 2.7.10) accompanied with several related statistics about the dictionaries produces (2.7.12). At the end of the section and the whole chapter, one more look at the results with the help the Vallex dictionaries is shortly discussed (2.7.13).

2.7.1 Evaluation data

The main part of the evaluation lies in comparing the extracted valency frames with manually annotated (gold) ones. For this purpose, I annotated manually 100 from PUD, because this treebank itself is annotated manually, not automatically, and the results measured on it are well comparable among many various languages, since the treebank contains always the same 1000 sentences in each language. I used Czech and English PUD, but there is not a Slovak PUD, unfortunately. Not all the sentences were used for the evaluation, only 100 of them, each tenth in particular (beginning with the sentence No. 10, ending with the sentence No. 1000).

2.7.2 Gold frames format

I have created the gold valency frames annotation according to the UD guidelines and my own intuition. For each sentence, I have described all valency frames I found, the verb and its arguments. In both cases, the ID of the token in the sentence is relevant and then some properties of the arguments, as described in this chapter (see 2.2.8): lemma for the verb and function (deprel) and form for the arguments. The form is described by the *Case* and *VerbForm* features and the forms of the argument's children with the *case* or *mark* deprels.The ID is important in order to ensure that the correct tokens are taken into account in case

that there were e.g. more verbs with the same lemma. Although the information about argument deprels and forms is mostly derivable from the token id, this does not need to be always true, because due to various syntactic transformations, the arguments can take other functions or forms than the basic ones, that are desired to be stated in the frame description. It is the task of the extraction program to revert these transformations and find the correct basic description of the argument. An analogous thing might hold for the verb lemma as well, although no lemma transformations are actually used anywhere in the program. The extracted frames are then converted into the same format and both sets of frames, the gold ones and the extracted ones, are compared. There may be also elided arguments added to the frame, which do not have a token ID.

2.7.3 Problems in data leading to multiple data versions

During the manual annotation, various problems came up. There were some indisputable mistakes in the data, other were less clear. There were also cases of consistent annotation violating the actual UD customs. I considered two ways to solve these issues, so that the tested program was not penalized for the mistakes: one option is to repair the mistakes in the data, the other is to keep them, but to create the gold frames accordingly. The former solution rises suspicion of data manipulation, the latter is inappropriate because of intentionally "incorrect" gold data annotation. In cases of individual mistakes I used the first solution, but I provided both the original and the edited data so that the changes can be checked and I perform the main evaluation on both sets for comparison (see 2.7.4). The second solution was used in cases of consistently inappropriate annotation in multiple places, because the intervention to the original data would be too big and also because we can assume that such annotation is present also on other places in the data, on which the evaluation is not preformed (see 2.7.5). Both sets of gold frames are provided on this case, too: the original gold frame and those adapted to the incorrect data annotation.

To sum up, there are two data sets and two gold frames sets, which make four different testing scenarios. I provide evaluation results only for two of them here for simplicity: the original (incorrect) data with the original (correct) gold frames and the partially corrected data with the adjusted gold frames (see 2.7.10).

2.7.4 Individual corrections in the data

The mistakes corrected in the data were often related to wrong deprels, upostags, heads or even lemmas. Their examples for Czech (73) - (78) and English (79) - (81) data are shown in the following sentences:

(73) "[...] bojoval dvě **hodiny** *[deprel: ~~obj~~ obl]* [...]"
= "he was fighting two **hours**"

(74) "[...] bylo shledáno, že se **dopouštěli** *[deprel: ~~ccomp~~ csubj:pass]* [...]"
= "it was found out that they had **committed**"

(75) "[...] používání [...] **se** *[deprel: ~~expl:pv~~ expl:pass]* nedoporučuje."
= "the use is not recommended" (reflexive passive)

(76) “[...] **se**[deprel: ~~expl:pv~~ expl:pass] [...] vláda rozpadla [...]”
= “the government fell apart” (inherent reflexivity)

(77) “Dostetek [...] **povede**[lemma: ~~povést~~ vést] ke zmatkům [...]”
= “enough of ... will lead[~~succeed~~] to chaos” (inherent reflexivity)

(78) “[...] jedna[id: 5] z kreativních **vedoucích**[upostag: ~~ADJ~~ NOUN, head: ~~5~~ 9, deprel: ~~amod~~ nmod] karnevalu[id: 9] [...]”
= “one of the creative **leaders** of the carnival” (originally verbal adjective, but here a noun)

(79) “[...] more babies named[upostag: ~~AUX~~ VERB] Keira [...]”

(80) “[...] temperature is above freezing[upostag: ~~VERB~~ NOUN] during the winter months.”

(81) “"We’ve requested other nations to help[id: 8] us[head: ~~10~~ 8, deprel: ~~nsubj~~ obj] populate[id: 10] the zoo [...]”

2.7.5 Consistent adjustment of golden frames

The changes in the golden frames take place in cases where the actual annotation of the treebank does not match the ideal state I had in mind when creating them. In Czech, I originally created golden frames for all verbal adjectives with a verbal lemma. There are actually two types of passive verbal adjectives: one closer to verbs that can be used only in nominal predicates (82), the other with completely adjectival endings that can be used also in attributes (83). Only the first type of passive verbal adjectives (and also the active ones (84)) bears the *VerbPart=Part* feature, which is necessary for distinguishing verbal and common adjectives (see 2.5.1). Therefore the frames based on the second type of passive verbal adjectives were removed from the golden annotation. However, the changes touched also the rest of verbal adjectives, because all of them have an adjectival lemma, while my original golden annotation contained verbal lemmas for them (*ozbrojit = to arm*, *bojovat = to fight*). In the modified version of golden frames, the adjectival lemmas are used (*ozbrojený = armed*, *bojující = fighting*), so they agree with the lemmas in the data. One more, less frequent correction was made: The verb *být* is annotated as copula (*AUX* upostag and *cop* deprel) if it forms a nominal predicate with an adjective (85). But in cases of nouns with prepositions (86) or of adverbs (87), it is annotated as full-meaning verb, while it should be still considered a copula. As this mistake is consistent among the data, I decided to adjust the golden data accordingly, so the valency frames are expected to be extracted here.

(82) “voják je ozbrojen” = “the soldier is armed”

(83) “voják je ozbrojený, ozbrojený voják” = “the soldier is armed, an armed soldier”

(84) “bojující voják” = “a fighting soldier”

(85) “Koncert byl[AUX, cop] hezký[ADJ].” = “The concert was nice.”

(86) “Koncert byl[VERB, root] v parku[NOUN].” = “The concert was in the park.”

(87) “Koncert byl[VERB, root] včera[ADV].” = “The concert was yesterday.”

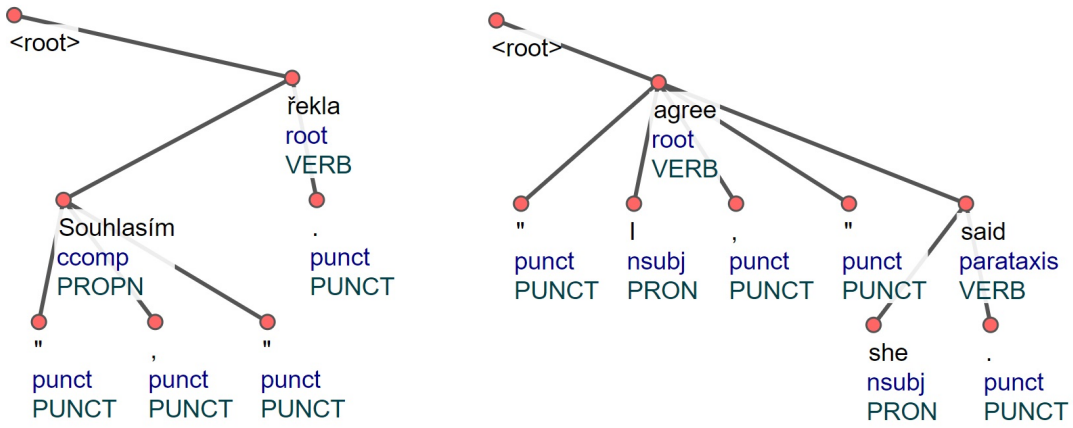


Figure 2.7: Comparison of Czech and English dependency trees for the sentences (88) and (89) with direct speech.

For English, such adjustments were made mainly for sentences with direct speech. According to the UD guidelines, the direct speech part of the sentence should be marked with *ccomp* deprel and depend on the verb of the introductory clause. This solution makes sense, because the direct speech is actually a replacement for the object that would the introductory verb have. The direct speeches are annotated this way in Czech data (88), but not in English. The situation there is reverse: the head of the sentence is the direct speech clause and the introductory clause depends on it with a special *parataxis* deprel (89), which is used also for various insertions, e.g. in brackets. Both solutions are visualized in the Fig. 2.7. In this situation I deleted the *ccomp* arguments from the gold frames for such sentences. Alternative means to handle this problem would be to creating an English extraction unit that would do the transformation of parents of *parataxis* verbs into their *ccomp*, but I decided not to go this way, because this approach violates UD guidelines and moreover in the new version of the English PUD (not used in this work), these cases are already annotated correctly; UDPipe also parses direct speech the correct way. Apart from this problem, various minor changes were also made, where my intuition differed from the actual annotation. E.g. I considered the word *armed* a verb and created a frame for it, whereas in the data (and also by the UDPipe), it is always marked with *ADJ* upostag, so there is no frame for it in the modified version of gold frames.

(88) “"Souhlasím*[ccomp]*," řekla*[root]*.”

(89) = “"I agree*[root]*," she said*[parataxis]*.”

2.7.6 Precision, recall and F_1-score

In the evaluation described below, the established precision-recall metric is used. It is appropriate in tasks, when there is a set of items and a subset of items meeting a specific condition is supposed to be chosen among them. The items actually meeting the condition are called relevant, the items chosen by the tested program are called selected. The aim is to develop such program, that the selected items fit

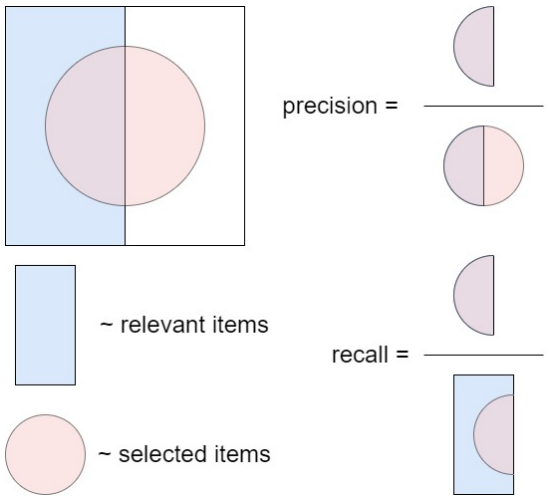


Figure 2.8: Scheme showing relevant (blue) and selected (red) items in a larger set including their intersection (purple) and how the precision and recall values are computed.

best as possible tho the relevant set. The values are computed from the two sets: the precision says, what part of the selected items is relevant (so choosing almost all items would lead to a low precision), the recall, on the other side, shows what part of the relevant items is selected (so not choosing almost any item would lead to a low recall). The illustration of how both values are calculated is shown in the Fig. 2.8.

F_1-score, the harmonic mean of the precision and the recall, computed according to the formula

$$F_1 - score = \frac{2 \cdot precision \cdot recall}{precision + recall},$$

is used as their appropriate combination, because it penalizes low values more then the arithmetic mean. In particular, both zero precision or zero recall lead to zero F_1-score. For the error analysis is important to pay attention to the individual values of precision and recall, but for the overall evaluation, it is sufficient to stated only the F_1-score, as it is done also here.

2.7.7 Evaluation scheme

Each extractor consists of several extraction units (see 2.3.3). Apart from the overall evaluation of the presented extractors, their units are tested also individually so they can be compared mutually. In order to ascertain the contribution of each unit, the results of the extractor including it are compared to a baseline extractor, which is nearly the same, but does not include the unit. For units of the main extractor, the base extractor is used as the baseline, for the language-specific units, the main extractor is used in this way. In the test of a unit, four extractors are run on the same data:

- BASE - pure baseline extractor without any extending unit
- FULL - full extending extractor with all its units

- PLUS - baseline extractor, but containing the tested unit
- MINUS - extending extractor, but without the tested unit

The contribution of the unit is measured in two ways: the results of BASE and PLUS are compared and the improvement is measured and similarly for MINUS and FULL (see Fig. 2.9). I will call BASE and MINUS reference extractors and PLUS and FULL tested extractors. For testing the whole extending extractor only one test is performed with FULL as the tested extractor and BASE as the reference one.

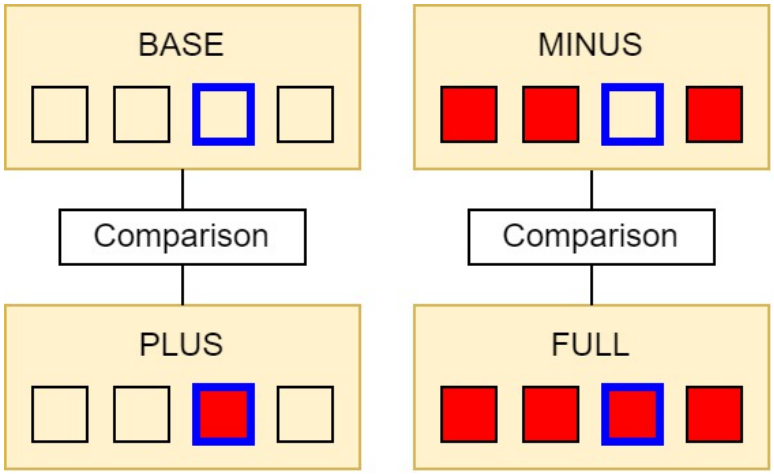


Figure 2.9: Overview of the evaluation of one extraction unit. The tested unit has a blue frame, the activated units are filled red.

2.7.8 Measured values

In the evaluation, four characteristics of the extracted frames are examined separately, so there is no one resulting number. The first two values are measured only for frame-level units, the other two are measured for both, frame-level and argument-level units (see 2.3.3 for the distinction). For each sentence of the evaluation data, the verbs of the extracted and gold frames are compared by their token IDs and the F_1-score is computed. Then, if the verb IDs agree, the agreement of the verb lemmas is checked and the portion agreements is computed. However, the results for lemmas do not vary between the reference and the tested extractor, because the golden frame lemma is the same as the lemma contained in the treebank annotation, from where it is extracted. I keep this value in the results because of the other version of golden data, where the lemmas are marked differently from the lemmas contained in the treebank annotation (see 2.7.1). In this case, this value does not reaches 100 %, because actually no unit does any transformation of lemmas. But it possible to implement such units, of course and then the value for lemmas would differ for the reference and the tested extractor.

If the verb IDs agreed a similar measurement is done on the argument level, too. The IDs of the chosen arguments are compared and the F_1-score is computed. Finally, for the agreeing arguments, their form and function (see 2.2.9) is compared and the portion of correctly described arguments is computed. An

argument pair receives maximum two points, one for the function and another for the form agreement. For the elided arguments which do not have an ID, the best match (gaining most points) is taken into account. The function and form it does not always directly correspond to the annotation of the chosen word, because during the extraction, various transformations may be performed. Unlike those for the lemmas, such transformations are actually done by the extractors in this work. These two argument-level values are shown for frame-level units, too, because the frame-level units may influence also the argument level. The argument-level values are averaged through all agreeing verb frames found.

To sum up, four percentage values are computed:

- verb ID - F_1-score of the verb IDs
- lemmas - portion of the agreeing verb lemmas (for agreeing verbs)
- arg ID - F_1-score of the argument selection (for agreeing verbs)
- arg desc - portion of agreeing argument functions and forms (for agreeing arguments)

These four computed values can hardly be combined into one, because they are computed in different ways (F_1-score vs. simple portion) and on different levels (frame level vs. argument level), so I decided to keep and always state the values separately. They are always measured for both, the reference and the tested extractor and the improvement is computed, as follows.

2.7.9 Improvement computation

Each value measured in the evaluation (see above) is represented by a percentage value, so each extractor lacks something to gain 100 % (at best 0 %). The improvement is computed as the portion of lack of the tested extractor corrected by the reference extractor, more exactly

$$improvement = 1 - \frac{tested\_lack}{reference\_lack}.$$

E.g. if the reference extractor scored 80 % (so lacks 20 %) and the tested extractor scored 95 % (so lacks 5 %), the improvement would be 75 %, because the tested extractor corrected three quarters of reference extractor's lack to 100 % (1 - 5/20). The same measure is applied even if the improvement is negative, so the improvement -50 % would arise if the tested extractor scored 70 %, not 40 %, which may look counter-intuitive at first, but it allows to compare tests with positive and negative improvements with the same measure (the percentage difference corresponds to the same amount of cases). Another related unpleasant consequence of this measure is that if the reference extractor scored 100 %, every worsening of the tested extractor would be infinitely negative, regardless of the absolute value of the score. The values presented are always calculated with two-digit precision and understood as percents (except the explained value -INF).

This improvement metric is often called also error reduction. Other ways to measure improvement are used too, such as absolute percent value

$$improvement = tested\_gain - reference\_gain$$

or relative percent growth

$$improvement = \frac{tested\_gain - reference\_gain}{reference\_gain}.$$

2.7.10 Evaluation results

In this section, the overall extraction results are finally presented and commented. The results presented in this section can be obtained via running test scripts included among the provided files. First results for the individual extraction units are given. Their complete evaluation was already presented in the sections describing them and here, in the Tab. 2.42 only their improvements are shown in order to compare them mutually. We can clearly see the improvement made in argument selection by the subject adding unit. But as the form is added in the language-specific extractors, this units worsens the value of argument description. Another worsening is visible in the oblique treating units on the Czech data, which was already discussed in 2.4.4, but even in English, the improvements are very low, which could lead to considerations of removing this unit from the main extractor. Language-specific extractors already include some units working also on the frame level and influencing the selection of verbs. On the other side, some only change various attributes of the extracted arguments. Most influential and helpful units are those treating the passive voice, which holds for both languages.

The overall results of the monolingual valency extraction are finally shown in the Tab. 2.43. The table contains evaluation for two versions of data and gold frames, as explained in 2.7.2 - 2.7.5. Here, we can see also some differences in the *lemmas* value. The results for language-specific extractors achieve higher values, as expected. Although the results for the adjusted data are better in general, the difference is not that big. Probably the most noticeable value is the extreme deterioration of the lemma agreement in the Czech extractor on the original gold data. The rounded value -2553 is caused by the properties of the metric used and by the differences in the annotation of lemmas of the Czech verbal adjectives, as described in 2.5.1 and 2.7.5. Another thing we can notice from the table are quite high values of the baseline extractor, which means most of the task is trivial. The only exception are forms of English arguments. Unlike pronouns, English nouns do not have the *Case* feature, it is added to the arguments by the language-specific extractor, where we can observe substantial improvement, but that is rather a technical issue. Another interesting observation is worsening of the argument description in the general extractor compared to the baseline (much more distinct in Czech). This is caused by the subject-adding unit, as explained already for the previous table.

2.7.11 Errors made by the extraction

After we have become familiar with the results, let us have a look at the errors the extractors have made. The error records are produced for each test run and stored

Main extractor (CS data)		subj		auxf			
	arg ID	74.05	73.31	-	-		
	arg desc	-80.95	-90.15	9.74	4.66		
		coor		oblq			
	arg ID	3.88	8.39	-2.78	-20.3		
	arg desc	3.67	4.13	-	0.69		
Main extractor (EN data)		subj		auxf			
	arg ID	48.12	53.52	-	-		
	arg desc	-9.94	-10.96	0.22	0.21		
		coor		oblq			
	arg ID	6.98	21.19	4.16	2.78		
	arg desc	2.49	2.29	1.67	1.04		
Czech extractor		vadj		pass		case	
	verb ID	86.3	86.3	-	-	-	-
	arg ID	-25.13	15.53	22.62	52.65	3.95	5.59
	arg desc	-19.89	-18.72	3.94	78.1	15.11	10.29
		nomi		pfin		numr	
	arg ID	-	13.48	-	-	-	3.02
	arg desc	76.28	87.44	1.2	11.71	2.41	22.5
English extractor		cvex		case		ptgr	
	verb ID	50.0	50.0	-	-	-	-
	arg ID	-	-	-	-2.19	11.34	32.92
	arg desc	-	-	72.75	91.53	1.95	41.93
		pass		that		cprt	
	arg ID	20.86	39.41	-	-	3.33	5.78
	arg desc	8.52	79.37	1.1	15.52	0.9	-

Table 2.42: Comparison of improvements made by individual extraction units for the measured values. The sign - indicates no improvement. No improvements are made in the *lemmas* value, so it is not shown at all. Most units do not improve the selection of verbs (*verb ID*). For each unit and each measured value, two improvements are given: the left one for the BASE-PLUS test, the right one for the MINUS-FULL test. Only units activated by default are shown here.

in the directory for logs (see C). Here only FULL Czech and English extractor (run on the adjusted data and golden frames) are examined. The examinations works with individual frame or argument occurrence. I divide the mistakes into six categories. There may be a verb frame missing (present in the golden frames, but absent in the extracted) or redundant (vice versa). If the frame is extracted for the correct verb, similar two errors can happend with its arguments. Finally, if an argument is correctly selected, it may have incorrect fucntion (deprel) or form (form and *case* and *mark* relations). The counts of these categories are given in the Tab. 2.44. As we can see, three of these error types are rare, so we will further deal only with missing or redundant arguments and with incorrect argument forms.

Among the missing or redundant arguments, obliques have the most prominent portion, as the Tab. 2.45 shows. Examples of redundant arguments are

		original data/frames			adjusted data/frames		
		BASE	FULL	impr	BASE	FULL	impr
Main extractor (CS data)	verb ID	87.94	87.94	0.0	91.24	91.24	0.0
	lemmas	99.43	99.43	0.0	100.0	100.0	0.0
	arg ID	78.69	94.26	73.06	79.11	94.43	73.34
	arg desc	83.88	72.48	-70.72	83.36	72.6	-64.66
Main extractor (EN data)	verb ID	97.78	97.78	0.0	98.66	98.66	0.0
	lemmas	99.55	99.55	0.0	100.0	100.0	0.0
	arg ID	68.62	86.18	55.96	69.47	87.39	58.7
	arg desc	45.34	42.16	-5.82	45.36	58.7	-6.13
Czech extractor	verb ID	87.94	95.79	65.09	91.24	98.8	86.3
	lemmas	99.43	84.88	-2553	100.0	100.0	0.0
	arg ID	94.26	95.96	29.62	94.43	96.21	42.37
	arg desc	72.48	95.53	83.76	72.6	97.21	89.82
English extractor	verb ID	97.78	98.43	29.28	98.66	99.33	50.0
	lemmas	99.55	99.55	0.0	100.0	100.0	0.0
	arg ID	86.18	92.34	44.57	87.39	93.48	48.3
	arg desc	42.16	96.21	93.45	42.01	96.08	93.24

Table 2.43: Overall evaluation of monolingual valency extraction. The results are shown separately for two variants of data and gold annotation.

simple to imagine - they are specifications of space or time. The opposite examples of missing oblique arguments are shown in (90) and (90). .

(90) "[...] barvy příslušející k daným kalendářním dnům [...]"
→ příslušet: [nsubj Nom] [**obl k+Dat**]
= "colors corresponding to the given calendar days"

(91) "[...] punish agents found to have engaged in unethical behaviour."
→ engage: [nsubj Nom] [**obl in**]

(92) "[...] he claimed to have*[Inf]* fought*[Part]* [...]"

Regarding the argument-form errors, in Czech, the most frequent are situations, when the case-adding unit did not manage to assign a case to a case-less argument or assigned a wrong one. Wrong conjunctions (*mark*) of *ccomp* arguments and incorrect verb forms of *xcomp* argument appear, too. Wrong verb forms and conjunctions of open clausal complements (which are way more frequent than in Czech) are the most frequent mistake done by the English extractor. One reason is that *xcomp* arguments were not included into the cases adding (so they are missing cases). The extraction unit could be easily extended in this way. Another situations are uses of perfect infinitives as clausal argument, where, because of the semantic priority of UD (see 1.1.8), the head of the construction is a participle instead of the infinitive (92). The solution could be simple here, too: creating an English unit, that would prefer the infinitive form of the auxiliary child to the participle form of the full-meaning verb itself (similar as the Czech `pfin` unit does with finite auxiliary forms).

error type	CS	EN
total	67	88
verb missing	4	1
verb redundant	1	2
argument missing	16	37
argument redundant	24	26
incorrect arg function	3	4
incorrect arg form	19	18

Table 2.44: Counts of various errors (error types) made by FULL Czech and English extractors.

error type	CS	EN
argument missing	44 %	59 %
argument redundant	79 %	62 %

Table 2.45: Portion of obliques among missing and redundant arguments.

2.7.12 Dictionary statistics

The monolingual extraction process produces a valency dictionary, which can be found in the relevant directory (see C). In this subsection, several basic statistics regarding the produced valency dictionaries extracted form Czech and English PUD are presented. They are taken from the FULL version of the dictionary i.e. extracted with all units that are activated by default. Counts of individual dictionary items are shown in the Tab. 2.46, while average portions among them are captured in the Tab. 2.47. From the first table we can see, that Czech little more diverse use of verbs and verb frames. This agrees with the statistics from the second table, that shows us English verbs as having more frames per verb (so where the Czech uses two distinct verbs, there are two frames of one verb in English) and more occurrences of a given frame type.

Following statistics (Tab. 2.48) present the ten most common argument description (deprel, form and *case* and *mark* relations). The counts given are computed from the argument occurrences. It is not a surprise, that subjects in nominative and objects in accusative lead the chart. Then the situation is more language-specific. Note that *expl:pv* on the Czech side and *compound:prt*

	CS	EN
verb records	942	739
frame types	1204	1104
frame occurrences	2022	2131
argument types	2688	2247
argument occurrences	4257	4110

Table 2.46: Counts of individual valency dictionary items.

	CS	EN
frame types per verb record	1.28	1.49
frame occurrences per verb record	2.15	2.88
occurrences per type	1.68	1.93
arguments per frame	2.23	2.04

Table 2.47: Relevat portions (givenn in %) between valency dictionary items.

on the English side are found among the argument rather from a technical reason (see 2.1.8). Interesting fact is that in subordinate clauses English slightly prefers using infinitives constructions to finite verb forms introduced with a conjunction, while in Czech, the situation is the other way around. Some obliques arguments manage to classify in the first ten, but as they represent the most interesting and complex part of this chapter, a separate table is shown for them: Tab. 2.49. However, the counts for them are already very low, so only first five oblique arguments are presented, because for lower counts the evidence would not be convincing. This table is different from the Tab. 2.11 shown earlier, which presented most common forms of oblique complements in the treebank before the extraction, without any decision about their inclusion into the valency frames. Some of the forms are present in both tables, but the most remarkable difference is the absence of the most common oblique complement form *v+[Loc] = in* among the oblique argument forms. This is because this preposition often denotes a place where an action is happening, which is not specific for one frame, but it can be used with many verbs. Finally, this observation means that the treatment of obliques during the extraction behaves meaningfully and the hardest problem of this chapter is acceptably solved.

	CS		**EN**	
1	nsubj [Nom]	1971	nsubj [Nom]	2104
2	obj [Acc]	960	obj [Acc]	1251
3	expl:pv	223	xcomp [Inf]+to	114
4	iobj [Dat]	174	ccomp [Fin]+(that)	107
5	ccomp [Fin]+že	84	compound:prt	70
6	xcomp [Inf]	76	xcomp	61
7	obl do+[Gen]	60	obl on+[Acc]	38
8	ccomp [Fin]	34	obl from+[Acc]	29
9	obj [Dat]	28	expl	27
10	obl [Ins]	26	obl for+[Acc]	24

Table 2.48: Deprels and forms of the ten most common valency arguments.

2.7.13 Matching extracted frames with frames in Vallex

Apart from the exact evaluation on the manually annotated gold data, I want to mention one more statistic that offers better insight into the extracted frames. I compare the extracted valency dictionary with the existing Vallex dictonaries for

	CS		EN	
1	do+[Gen]	34	on	38
2	[Ins]	23	from	29
3	k+[Dat]	20	for	24
4	na+[Acc]	19	as	19
5	po+[Loc]	17	at	19

Table 2.49: Forms of the five most common oblique arguments. All of the English obliques would be in accusative case, so this is not shown.

Czech and English (see 1.4.5) and compute how many frames have the two dictionaries in common. Comparison with CzEngVallex is used in the next chapter as the main means of evaluation, therefore the matching process itself is described there (see **??**). Comparison with Vallex cannot be used for evaluation here, because the extraction is based on the given data, which may actually contain a different set of valency frames than there are in Vallex, so the unmatched frames on either side cannot be directly considered mistakes.

Two basic types of disagreement can possibly appear. We might extracted a valency frame that is not present in the Vallex or some frame contained in Vallex is missing in our dictionary. Both cases could be a mistake, but not necessarily; there may be legitimate disagreements between the Vallex frames and the extracted frames. The second case happens when a Vallex frame is not present in the data so it cannot be extracted. On the other side, there are many verb frames missing in Vallex, causing the first case of legitimate disagreement. The only output from this experiment are portions of frames from the extracted dictionary present in Vallex and vice versa (but the latter value describes rather the richness of the data rather than quality of the extraction).

The results of the monolingual matching with Vallex dictionaries are summed up in the Tab. 2.50. Each extracted dictionary is compared with the separate valency dictionary for the respective language (Vallex for Czech, EngVallex for English) and with the corresponding part of CzEngVallex. Counts of lemmas in both compared dictionaries are given with their intersection. Frames are matched and compared in a similar way, but only frames of the preiously matched lemmas are taken into consideration (so the total number of frames means total of the matched lemmas, so do the percentages). Two matching strategies are used: The first one matches the frames uniquely (each frame can be matched only once), so the number of pairs in the same as the number of matched frames on both sides (and therefore only percentages are given). The second approach allows multiple possible matches, so the number of pairs is higher than the number of matched frames on each side. The matching strategies are discussed and compared in more detail in the next chapter.

The table shows that the Vallex dictionaries cover most of the verbs appearing in PUD, but only less than a half of their frames, although this may be caused by the errors of the matching algorithm. On the other side, it shows that majority of the verbs and frames in Vallex did not appear in the treebank, because PUD contains only 1000 sentences. Of course, the strategy allowing multiple matches has a higher coverage, more significant on the Vallex side.

		CS PUD extr	pairs	Vallex
lemmas	total	942		3595
	- matched	66,5 %	656	18.2 %
frames	total	907		2971
	- matched (unique)	45.5 %	413	13.9 %
	- matched (multiple)	422 ∼ 46,5 %	901	797 ∼ 26.8 %
		CS PUD extr	pairs	CS CzEngVallex
lemmas	total	942		11639
	- matched	79.5 %	749	6.4 %
frames	total	994		3402
	- matched (unique)	40.6 %	404	11.9 %
	- matched (multiple)	415 ∼ 41.8 %	1336	1227 ∼ 36.1 %
		EN PUD extr	pairs	EngVallex
lemmas	total	739		4512
	- matched	92.0 %	680	15.1 %
frames	total	1043		2346
	- matched (unique)	43.3 %	452	19.3 %
	- matched (multiple)	529 ∼ 50.7 %	1174	902 ∼ 38.5 %
		EN PUD extr	pairs	EN CzEngVallex
lemmas	total	739		4351
	- matched	91.6 %	677	15.6 %
frames	total	1038		2302
	- matched (unique)	43.2 %	448	19.5 %
	- matched (multiple)	527 ∼ 50.8 %	1145	887 ∼ 35.5 %

Table 2.50: Agreements in lemmas and frames between the valency dictionaries extracted from PUD and Vallex dictionaries. Only frames of the matched lemmas are taken into consideration. Two frame-matching strategies are used: one matching each frame only once, the other allowing one frame to be matched with several frames on the other side, so the number if frame pairs is higher than the numbers off matched frames on each side.

Chapter 3

Cross-lingual valency frames linking

This chapter is about linking of valency frames and their arguments between two previously extracted valency dictionaries. It begins wiht several introductory and explanatory paragraphs. After that, two basic approaches for the linking are examined and evaluated. The chapter is closed with an experiment trying to use the produced valency dictionary for a more advanced task.

3.0.1 Monolingual extraction vs. cross-lingual linking

Building a multilingual valency dictionary, which is the task of the program presented in this work, consist of two main steps. The first one was the extraction of monolingual valency frames, described in the previous chapter (see 2). Once the frames of different languages are extracted, the time comes for linking of corresponding frames across the languages as the second main task, which is the topic of this chapter.

3.0.2 Bilingual and multilingual extraction

Although the ultimate task is to build a truly multilingual valency dictionary for any number of languages, the linking methods are primarily focused on two languages as the simplest case. Building a trilingual valency dictionary for languages A, B and C can be realized by running two linking processes separately, i.e. by building bilingual dictionaries AB and BC and then joining them via the valency frames in B. This is dealt with on the level of visualization of the dictionary: for each A valency frame, several B valency frames could be displayed and for each of them yet multiple C frames may follow. If we build the third valency dictionary AC, a question would arise, how to visualize that result. The same holds for more than three languages: such multilingual dictionary can be built by building a series of bilingual dictionaries and then joining them for the visualization.

3.0.3 Basic approach: parallel corpora vs. language similarity

The typical and most general case, at which this work is aimed, is to use a parallel corpus. Various methods for linking valency frames in a parallel corpus are described in 3.1: word alignment method, sentence structure method and

language similarity method. The last mentioned approach assumes that the two languages are similar and tries to take an advantage of it.

3.0.4 Parallel corpus vs. dictionary

There could be another, maybe more natural option, where we had monolingual corpora and a bilingual dictionary, which served us for verb linking instead of relying on language similarity in the previous case. This option aims for more universality in comparison with the first one. However, obtaining a dictionary for a chosen pair of languages is similarly difficult as obtaining a parallel corpus; actually for many language pairs the dictionary is obtained from the parallel corpus. Moreover, in that case we lose the grammatical information and alignment contained in the parallel corpus, so this option is not further discussed in this work.

Similarly as in the monolingual extraction, where the program had to find the verbs first and then their arguments, the bilingual frame linking can be also divided into these two substeps: linking the correct verbs and linking their arguments.

3.0.5 Relation between extraction and linking

The processes of valency frames extraction and linking, as they are implemented in this work, happen separately, so the results of the linking do not influence the extraction. There are options to expand the solution, e.g. so that the extractors decided the inclusion of some obliques based on the successful inclusion of their counterparts in the other language, as indicated in 2.4.5. The separation allows running the linking after the extraction. If we do not need to save the intermediate monolingual dictionaries and are only interested in the resulting bilingual valency dictionary, it would be possible to perform both tasks simultaneously given a parallel treebank, i.e. extract frames from two parallel sentences and link them right after that, unless there was a postprocessing at the end of the extraction. But since there are several language-independent extraction units that take place after the whole monolingual dictionary is created (see 2.4), it is not possible to do the linking during the extraction and we always have to first finish both monolingual extractions and only then run the cross-lingual linking.

3.0.6 Terms linking vs. matching

At the end of the introduction to this chapter, let me explain the notions of linking and matching as I use them in this chapter. Both describe connecting of two items, but in different contexts. I refer to connections between valency frames or arguments of two different languages, which is the main topic of this chapter, as to links, I call the process of establishing the links linking and the program objects performing the linking are called linkers. In all other context I use the word matching, especially when connecting the extracted valency frames and arguments with their counterparts in CzEngVallex. The part of the program searching the best matching is called matcher. However, it is used also by linkers. In fact, the linkers assign values to various item pairs, the matcher finds the best matching (exclusive subset of the pairs) and the linkers then create the link.

3.1 Valency frames linking methods

This section presents various methods for linking valency frames in a parallel treebank. We suppose the corpus is already sentence aligned. If not, the sentence alignment must be preformed, which should not be a difficult task. The parallel corpora used in this work fulfill this assumption, so this step is not further discussed. We must remember, that we do not select valency frames anymore – they have been already selected from the monolingual valency frames extraction. It means that the valency frames might not correspond to each other precisely, not only because of the properties of the languages (which is correct, this is exactly the thing we want to capture in the valency dictionary), but also because of inconsistent treebank annotation or mistakes in the monolingual frames extraction (see 2.7.11).

I come up with three main approaches for linking of valency frames, which will be further combined. In order to measure the performance of the presented methods, it is necessary to set up a baseline solution which links verbs and arguments in a trivial way, so that the improvement of the other methods could be computed in comparison with the baseline (see 3.1.3). The first method uses word alignment of the corresponding sentences in order to link the verbs and their arguments (word alignment linker; see 3.1.4). The second approach relies exclusively on the similarity of the morphological and syntactic information contained in the UD annotation of the sentences (sentence structure linker; see 3.1.5). The third option is to use the similarity of the individual words on both sides (language similarity linker; see 3.1.6).

3.1.1 Linking algorithm

Similarly as there were extractors for the extraction task discussed in the previous chapter, the class dealing with linking valency frames and valency arguments is called *linker*. Each advanced linker is inherited from a common ancestor class `Linker` and implements analogical methods for the levels, on which the linking takes places: linking verbs and linking valency arguments. In both cases there are two sets of items on the input, either verbs in the parallel sentences or arguments of chosen valency frames. Let us say, there are m items on one side and n items on the other. The task is to link the corresponding items, in other words to find their correct matching. Since m and n may differ, the matching does not need to be perfect, i.e. some items will remain unlinked. There is $m \cdot n$ possible item pairs and the linker tries to choose the best ones among them. In order to do that, it assigns a score to each pair, creating a table of size $m \cdot n$, filled with the scores of the individual pairs. Individual linkers implement different methods for computing the score for two given items (verbs or arguments).

There are many sentences with only one item on both sides. Linking between such two sets seems trivial and without need to compute the score. Still, a low value of the score can point out that the sentences do not correspond to each other (e.g. because of a very loose translation or an error in the sentence alignment) and allows us not to accept the pair. Therefore the score is computed even in this case. There may be a threshold for the score specified in the scoring methods; if the score does not achieve the threshold, it is replaced with a special

value prohibiting the given pair. This solution should help not to incorrectly link remaining items.

After filling up the whole score table, the best combination of the pairs based on their scores is found, which may not be a simple task (see 3.1.9[1]). Each pair of the chosen list of best item pairs is linked together. If the linking was performed on the frame level, then argument-level linking follows for each linked frame.There could be a strategy first to link only verbs and then, after processing the whole treebank, to use the information from the verb linking to improve the linking on the argument level,[2] but this idea is not expanded in this work. Another possible heuristic could lie in rejecting a frame link based on the non-correspondence of their arguments, so the argument level would influence the frame level. This idea slightly appears in the sentence structure linker, which takes into account the number of verb's children, but the algorithm implemented in this work for simplicity strictly separates the linking on both levels. Because of the fact that the linking process for frames and arguments is similar, the methods are described only once, but the verb linking and argument linking is evaluated separately later.

3.1.2 Linker parameters

All the presented linkers (except the baseline) depend on various parameters influencing their work. Some of them are binary, other are numerical. Beside of the parameters specific for their work, each linker has the already mention threshold needed for an item pair score to be achieved, so that the pair could be linked, which is also a kind of parameter.[3] The combined linker has weights of the individual linkers as their parameters. We need to tune the parameters to achieve the best performance of the linkers. The tuning of the parameters does not happen for each linker separately, but for all subordinate linkers of the combined linkers together. This process and its results is described together with the evaluation in **??**.

3.1.3 Baseline linker

The baseline linker performs linking in a trivial way in order to measure the true contribution of the various advanced linking methods. It links verbs or arguments according to their order in the sentences, i.e. the first item on one side with the first item on the other side, similarly for the second item etc. If there are more items on one side than on the other one, the overhang will remain unlinked. At least on the frame level, this method is motivated by the idea, that the concepts in complex sentences are often expressed in a similar order, so the order of the used verbs could correspond. This holds less on the argument level, because the word order of a simple clause is more diverse among the languages than the order of clauses within a complex sentence, although they primarily differ in the placing

[1]The algorithm for that is described in the evaluation section, because it is used also for a completely different task in the program - matching extracted valency frames with the frames from vallex dictionaries

[2]This was done for some extraction units of the main extractor, that implemented postprocessing parts, where they compared the extracted frames with each other a based on that they decided on changing them

[3]actually, it would be even more correct to have two such thresholds - one for the verbs and the other for the arguments, but this is not implemented in this work.

of the verb rather that the arguments (the three most frequent argument orders of bivalent verbs are SOV, SVO and VSO, where V = verb, S = subject, O = object[4]), so the idea is not totally meaningless here, too.

3.1.4 Word alignment linker

The first non-trivial approach to linking valency frames uses word alignment between the two parallel sentences. This assumes, of course, that the word alignment was actually performed before the linking (which is done by Fast_align in this work, see 1.4.7 and E.1).[5] Theoretically, if the alignment is done correctly, it should hold that verbs and their arguments are aligned with their counterparts in the parallel sentence. In reality, the situation is much more complicated due to compound verb forms, pronoun dropping and many other linguistic phenomena making bijection between words in parallel sentence impossible.

The word alignment itself is not so straightforward either. Most of the tools, including Fast_align used in this work, do the word alignment one-directionally: for each word on one side they search the best counterpart on the other. Not all the words on the first side need to be paired, but no word from the first side will be paired with multiple words on the other. For a symmetric alignment we need to run the aligner two times, once for each direction, and then somehow combine the results. Simple combining methods are union and intersection of the one-directional alignments. The union leads to ambiguous word alignment, because multiple words from one side can be assigned to a single word in the other language, so that further decision is required (especially when the reverse alignment assigns their common target to none of them). Usage of the intersection means loss of alignment, which was possibly correct, but was found only in one direction. Probably the best option is to combine these two methods, e.g. to prefer the two-directional alignment of the intersection, but secondarily look at the rest of the alignments that are left in the union results.

There are more possibilities for the combination depending on how much the two-directional alignments are preferred over the one-directional ones. Several of them are tested in this work and are presented in the following tables. They assign 1 point to a word pair corresponding to a two-directional alignment and they differ in the score assigned for a one-directional alignment. Several values are used for tuning of this parameter in 3.2.4.

3.1.5 Sentence structure linker

A completely different strategy for valency frames and arguments linking is to use the morphological and syntactic information of the contained in the treebank annotation. Although languages have different grammar, so the dependency trees and annotation of their sentences also differ, some of the relevant characteristics might be shared. This linker looks at the similarity of chosen features in both parallel cases. The approaches to the verb linking and the argument linking differ significantly here, because many of the features relevant for verbs cannot be used in the case of arguments. The desired features are defined separately

[4]Dryer and Haspelmath 2013, `https://wals.info/feature/81A`

[5]It does not need to be done if we know this type of linker will not be used, e.g. it is deactivated in the configuration file.

for both levels. During the run, the linker compares different pairs of verbs or arguments and for each pair it calculates, how much of the relevant grammatical characteristics they share.

For the frame level, I selected six features to be taken into account by this method, each of them contributing to the score for a given verb pair or argument pair. They are either binary (thus contributing 0 or 1 to the link score) or numerical (described by a natural number). In order to have the possibility to compare these parameters and adjust their weights, the numerical ones are transformed into a value from 0 to 1. All of them are differences between numerical features of the verbs (order or depth in the sentence, number of children), so they favour the linkage more, when they are small (i.e. the numerical features do not differ so much). Later we will want to maximize the link score, so inversion is a good solution here: lower values are inverted into values close to 1, higher values close to 0. In case the original value of a numerical parameter is 0 (thus not possible to invert), the inversion is calculated as

$$score = \frac{1}{a+1},$$

where a is the original value of the numerical parameter, which gives 1 for the most favouring features with original value 0 (saying that the numerical features of the items do not differ at all), and descends with growing original value.

All the features should show whether two given verbs correspond with each other. Binary features capture the agreement of key categories between the verbs. As we take only nodes with the *VERB* upostag into account, there is no need to check the agreement on the upostag of the verb nodes themselves. But the upostag of the parent can be a relevant feature, so can be also the deprel of the verb nodes to their parents. The deprel of the parent seems already too distant, so do these characteristics of further ancestors, so their differences would say less about the pair. The deprels and upostags of the children cannot be chosen as a feature, because their number can differ. The individual UD features can differ among languages, so we cannot take the agreement on a value of a particular feature into account if the feature is not annotated in the given language. So there are two relevant binary features: the agreement of verb node deprel and the agreement of parent node upostag.

The numerical features capture the difference in various relevant values. We can measure the linear order of the verb node or of its parent node in the sentence. We can measure the depth of the verb node in the sentence tree. The parent node has the depth always one lower, so it makes no sense to take its depth as a separate feature. The last feature considered is the difference in the number of children.

To sum up, the features used for verb linking are:

- difference between verb parent node linear order (numerical)
- verb parent node upostag agreement (binary)
- difference between verb node linear order (numerical)
- verb node deprel agreement (binary)

- difference between verb node depth in the sentence tree (numerical)
- difference between number of child nodes of the verb node (numerical).

If the verb is the root of the sentence, it has still a technical root as its parent, which has linear order equal to zero and upostag `<ROOT>`, as defined by Udapi, so there is no need to special treatment of these cases.

The discussed six features are relevant for the linking on the frame level. Regarding the argument linking, they are all irrelevant. Those examining the parent node are not relevant, because the parent node is the same for all considered arguments, as the argument linking takes places after the verbs have been linked. Similarly the linear order and the depth in the sentence depends already on these characteristics of the verb. The depth is usually one level deeper than the depth of the verb. We could include the distance of the argument to the verb, but this is something that various language could have different even for the corresponding argument. The correspondence between arguments with different deprels and morphological features (because arguments are defined by these) is the interesting output of this work, so the their agreement should not be used for the linking. Finally, the number of the children is not very relevant either for the linking of the arguments. The only feature used for the linking on the argument level is the upostag of the arguments.

3.1.6 Language similarity linker

The third linking strategy is based on language similarity, so it is not language independent. It is based on comparing the characters of the words to find the best pairs to link. We hope that linking similar looking verb lemmas also captures the correspondence in meaning, at least partially. It is intended to be used with closely related languages, where it may help a lot, while its use with unrelated languages should lead to very bad results. The comparison of its work on such two pairs of languages (Czech-English as unrelated and Czech-Slovak as related) might be an interesting experiment.

For comparison of two strings, various similarity metrics can be used. Two simplest of them are implemented in this work: Levenshtein distance and Longest common substring. In addition to these two universal metrics, a specific Czech-Slovak version of the similarity linker is discussed in following paragraphs .

Probably the best known string metric is the Levenshtein distance, which grows by one point for each difference in the word. The differences accepted are insertion and deletion for cases, that a character on one side does not have a corresponding counterpart on the other and vice versa, but also substitution, which means replacing a character with another one. Especially this case is very frequent, because it reflects cases, when a phoneme in the common ancestor language developed into two distinct phonemes in the two descendant languages.

There are also other possible metrics. A distance based on the longest common subsequence (LCS) is also included in several tests. This metric is similar to the Levenshtein distance, but it does not allow substitutions. Therefore these type of differences must be modeled with insertion and deletion with the LCS distance, getting two distance points instead of one. This is a disadvantage, the substitutions are very frequent, as already mentioned. We must be aware that in

the original task of finding the longest common subsequence, the common letters are counted, not the differences. We are actually interested in the rest of the characters: that are not part of the LCS. The value this opposite metric is the the number of characters left after removing LCS (sum of such letters on both sides) There are also other metrics, using e.g. transpositions (called metatheses in linguistics), but as they are not very frequent, they were not tested.

Both the metrics mentioned work on a simple algorithm:

Algorithm 2 DISTANCE

```
1: procedure DISTANCE(A, B)
2:     if |A| == 0 or |B| == 0 then
3:         return |A + B|
4:     if A[0] == B[0] then
5:         return DISTANCE(A[1 :], B[1 :])
6:     else
7:         return 1 + min{
8:             DISTANCE(A[1 :], B),
9:             DISTANCE(A, B[1 :]),
10:            DISTANCE(A[1 :], B[1 :]) (only for Levenshtein, not for LCS)
11:            }
```

The input of the algorithm are two words, two verbs in our case, the output is their distance according to the metric. To find a closest counterpart for a given verb on the source side, we should apply it on all the verbs on the target side and then chose the one with a minimum distance. There can be several target verbs with the same minimum distance from the given source verbs, so the algorithm does not necessarily provide an unambiguous result.

This basic algorithm is actually very slow, because it solves every possible branch to the end, even if there is no chance to improve the minimum distance, or because it runs multiple times on the identical state (e.g. application of deletion and then insertion leads to the same state of the computation tree as the reverse application and as the substitution alone; the algorithm then continues the same way three times from this point). Also the application of the algorithm on all the pairs with the target verbs individually takes a lot of time.

To speed up the algorithm, it was improved with several heuristics. The main improvement is that the function gets the minimum distance found up to now and actual distance of the original strings as the parameters and that when the actual distance is greater than the minimum, the branch of the computation is terminated prematurely. The same happens if the actual distance plus the difference of the lengths of the strings left is greater then the minimum. This heuristic is applied even if the minimum comes from a comparison with another target verb.

I also tried a variant with parallel comparing with all the target verbs using a minimum heap. This would guarantee that the number of steps needed for finding the minimum distance is minimal, as the heuristics still allow for the first verbs to take steps taken, that later turn out to be unnecessary, as the minimum is actually lower than what it appeared to be by the time these verbs were being considered. However, the solution with the minimum heap required storing of all

the states of the computation from all the verbs in the heap, which showed to be too demanding for the memory space. This implementation was not included in the tests.

Another improvement is to sort the target verbs so that the verbs beginning with the same letter as the source verb would come first (and so that the minimum would be found earlier speeding up the other runs of the algorithm). This heuristic is language-dependent though, as some languages (such as Swahili) are characterized by intense usage of prefixes and the root (which is expected to be deciding for the similarity) comes at the end. It would be better to sort the target verbs according to the last letter in this case. However, for these languages it would be better to apply the whole algorithm reversely from the ends of the verbs to reach the maximum effect of the heuristics described. This is possible to implement in a language-specific module, but the general algorithm supposes, that it is more appropriate to start from the beginning (because suffigation is generally more common that prefigation). Thus the sorting by the first letter is kept in the general algorithm, too.

There are two parameters of the similarity linker. One decides, whether the substitutions are allowed or not, i.e. which of the two metrics should be used. The second parameter is not a true parameter of the similarity linker. It is applied when the linkers are created and decides, whether the language specific similarity linkers could be used or not. If yes, its use is decided according to languages specified in the command line (see E.1). There is a Czech-Slovak and a reverse Slovak-Czech similarity linker provided; they are described below. Because they redefine the price of the substitutions for several selected letter pairs, there is no point in using them with the substitutions deactivated. All they do is that they define language-specific substitutions and assign them a lower price than 1. That means, that a specific substitution does not increases the distance so much, because it is a regular substitution between the languages, so the linker should favor linking the two words.

3.1.7 Specific Czech-Slovak similarity linker

We can achieve better results in the task of linking verbs between two similar languages by defining some language-specific phonological changes that occur regularly or typically between the languages. The basic idea is to define pairs of such differences with some lower penalty than 1, so that a substitution modeling this pair would score better and leads to a lower minimum.

A problem appears, when such substitution involves more letters than exactly one on each side, either as a proper part of the difference or only as a part defining context, e.g. if the substitution $b \leftrightarrow p$ happens only after another, common letter m, we would like to define a pair $mb \leftrightarrow mp$ or in an actual example Czech t corresponds to Slovak $ť$ especially at the end of the verb (because it is the infinitive ending). However, including these multi-character pairs is complicated to implement. Apart of the actual substitution, the algorithm would have to store preceding letters in a buffer for some time in case they would make a substitution pair later together. This would lead to a greater branching, so that the algorithm would be very slow. Because of this, another solution was used instead of the buffers. All the possible multi-character pairs are firstly used to pregenerate a list of modified variants of the source verb (remembering the cost of the substitutions

used for each variant) and then the algorithm is applied to each such source string. This solution is easier both for implementation and for the computation. The mono-character substitutions are still treated in the algorithm itself, because their number is expected (and for the tested Czech-Slovak pair actually is) higher, so the set of pregenerated forms would be very large, and on the other hand they are simple to implement in the algorithm unlike the multi-character ones.

Similar idea could be considered also for the other two Levensthein operations, so that the cost of characteristic insertions and deletions of multi-character sequences would be lower than their length. However, no such sequences were considered for the Czech-Slovak language pair.

Substitutions preferred between Czech and Slovak are divided into three levels according to their reliability. The substitutions of infinitive endings *t$* ↔ *ť$* and *ci$* ↔ *ct$* (where *$* stands for the end of the string) are quite sure and thus cost 0. Most of the other defined substitutions have cost 0.2. Many of them include palatalized consonant on one side, while its hard counterpart on the other, differences in vowel length or other typical substitutions (*ů* ↔ *ô*, *ou* ↔ *ú*, *ě* ↔ *ie*, *z* ↔ *dz* etc.). The third level includes all possible substitutions of vowels and diphthongs uncovered in the previous set with the cost 0.5. This rule is based on the observation, that vowels changes are easier to happen during the language development than consonants, so that a substitution *a* ↔ *e* should be more probable than *a* ↔ *m*. I chose the costs for these language-specific substitutions (0, 0.2 and 0.5) according to my intuition as a speaker of Czech and Slovak; there is no statistic computation leading to them. The choice reflects the certainty I think the substitution can be applied.

3.1.8 Combined linker

As the mentioned linking approaches are based on completely different types of information, there is a high chance that we can obtain the best results by their combination. This can be well realized by combination of scores from the individual methods when considering a given verb pair. As well as other linking methods have their parameters, the parameters of the combined linker are the weights in the combination.

3.1.9 Finding optimal matching

The following paragraphs describe an algorithm used in various parts of the program. There are two sets of items of sizes m and n given on the input and a score table of size $m \cdot n$ describing scores for each item pair, in other words a weighted bipartite graph. The task is to find the maximum matching between the items. Generally, the matching cannot be perfect, because m and n may differ, so some items will remain unmatched.

To find the truly best matching I tried to used linear programming (with help of Python's `PuLP` library), but the efficiency of this approach shows to be insufficient. Therefore I decided to choose a faster, but suboptimal solution, my own heuristic. It lies in trying all combinations to a certain depth and than continuing in a greedy way to chose. The first phase runs a recursion for each pair of linkable items, if the given depth was not reached. An item pair is linkable if none of the items has already been already chosen into a link with another item

$m \cdot n \leq 100$	depth = 3
$m \cdot n > 100$	depth = 2
$m \cdot n > 1000$	depth = 1
$m \cdot n > 10000$	depth = 0

Table 3.1: Value of the critical depth dependent on the product of number of items on both side.

and if there is a score given for the pair (because there could be `None` in the table preventing the link at all). In the second phase no more branching is done and from the rest of items the pair with the highest score is always chosen until there is no item left on one of the sides. The critical depth is dependent on the numbers of items. The higher the depth is (so the deeper the first phase goes), the better should be the result, but the slower is the computation. If the items are few, the depth is high, in the opposite case it is low. More precisely, the depth is computed according to the product of m and n, as the Tab. 3.1 shows.

This algorithm is used at multiple places in the program. It is applied on extracted frames to match them with frames from Vallex dictionaries. This was already used in 2.7.13 only for statistical purpose, but it is used also in 3.2 for the evaluation. Apart from matching with Vallex frames, it is applied also for the valency frames and argument linking. This happens in 3.1.1 for individual sentences in the parallel corpus (for linking once frames, another time arguments as items). It is used for all three linkers. It would have been more appropriate to search minimum weighted matching for the similarity linker, as it works with distances, i.e. negative score, but to unify the approach to all linkers, the similarity linker was modified so that each score is inverted (see 3.1.6), so that the usual algorithm for maximum matching can be used for it as well.

3.2 Frame linking evaluation

3.2.1 Evaluation on gold data vs. comparing with CzEngVallex

Unlike the frame extraction, which was evaluated only on data with gold annotation (see 2.7), the frame linking is tested in two ways: on manually annotated frame pairs, too, but also by comparison with CzEngVallex (see 1.4.5). The evaluation on gold data is more reliable, because the annotator can precisely describe the tokens (by their IDs), so the program can match them properly, while the automatic matching with frames in an existing dictionary may be erroneous. On the other hand, the annotation is a time-consuming task, so the size of the gold data provided is small in comparison with CzEngVallex, where many verbs and frames can be found, all the more that the vallex dictionaries themselves are manually built, so they are a high-quality resource. Apart from the proper resource used for the evaluation, both strategies use different metric. Both these ways of assessment of the linking system are described later in more detail.

3.2.2 Manual annotation of PUD

Regarding the gold evaluation, PUD is used. The manually annotated frames used in the previous chapter for the evaluation of the monolingual extraction are linked here as the gold annotation for the linking task. Every tenth sentence was taken from PUD into the evaluation set. In both cases, 100 sentences were annotated and used for evaluation, which is rather a small set. The bigger the evaluation set is, the more reliable the results are. For JRC Acquis, 1000 of sentences were randomly chosen from the whole corpus (which has 940,427 parallel sentences in the case of the Czech-Slovak alignment), but only the first 100 of them were annotated.[6]. In the end, there are 100 parallel sentences used for the gold evaluation from the PUD treeeach treebank. Note that for the vallex evaluation, no such selection set is needed; all the data from the corpus can be used.

3.2.3 Cross-validation

Apart from the final evaluation itself, we need also development testing data for tuning various parameters of the linkers. These two tasks should not be performed on the same data, otherwise we would risk overfitting the system to the data and the measured results would not indicate its real performance on new sentences. On the other side, the set of annotated data is small, so it would be good to fit the parameters of the final linking system on all of them. So I suggest using cross-validation, which is a well-established evaluation strategy. The gold data set of 100 sentences is divided into four parts, called folds. Various numbers can be used with the cross-validation, but for such small data sets like in this work, there must be rather a small number of folds. The testing is run four times and always three fourths of data serve for fitting the parameters, while the last one for the evaluation. In our case, each fold will have 25 sentences. The results from all four runs are then compared and their average is considered the final result of the system. If they are similar, it indicates, that the evaluation is probably sufficiently robust to give meaningful result of the tested system. The more they vary, the lower evidentiary value the evaluation has.

Unlike the extractors, the linkers do not do any postprocessing of the linking based comparison of all links across the whole dictionary, but each sentence is linked separately, so it is sufficient to run the gold evaluation on the annotated sentences. The vallex evaluation, which does not need any annotation, can be run on all the data, but still needs to use the cross-validation in order to properly tune the parameters. The number of folds for the vallex evaluation could be higher, because dividing one thousand or one million of sentences would give enough sentences for the evaluation in a fold even for many folds, but for simplicity, I use a 4-fold cross-validation also in this case.

3.2.4 Tuning the parmeters

The linkers presented in 3.1 have various numeric parameters that influence their performance. The parameters themselves can be binary or numerical and the numerical differ in the range of values that are meaningful for them to acquire. In

[6]The list of the indices of the 1000 sentences is provided in

order to optimize the choice of the values for the parameters, various combinations of values are generated and a linker is created for each combination of the values. The parameters of the linker, that achieves the best value, are considered the best. This is done separately for all three main linkers. After that, the process is run one more time with the combined linker, where the weights of the subordinate linkers serve as the tuned parameters. This way the parameters for best linker of one cross-validation run are tuned. The chosen linker is then evaluated on the remaining fold, that was not used for tuning. The parameters of the final linking system, are tuned the same way, but on all the data, without any further evaluation.

3.2.5 Overview of the linking process

To sum up, the evaluation process consists of four runs of the cross-validation, each run consisting of tuning the parameters (first for the main linking methods and then for their combination) and then the evaluation itself. This whole process is run for several times. The Czech-English linking is evaluated twice: once with the gold evaluation, the other time on CzEngVallex. Moreover the whole described process takes places twice, separately for frames and arguments.

3.2.6 Evaluation metric

Both means of the evaluation (the gold one and the CzEngVallex one) are set into the same situation. They get two sets of extracted frames from parallel corresponding sentences and a set of links the linker created. The evaluator decides, which links are correct, which are not and which are missing. They decide it based on their own idea of how the frames should be matched. But we must be aware that the sets, from which we are choosing need not to be the same. The CzEngVallex is a dictionary and first we have to match the extracted frames from the sentence with some of the many frames in CzEngVallex. It can happen that no such frame is found. And certainly vice versa, most of the frames in CzEngVallex will not be present in the evaluated sentence. The situation in gold evaluation is somehow different, because the frames can be precisely matched via indices in the sentence. But as the annotations were made regardless to the extraction, so some frames present on one side may be absent on the other.When the linker does not have any chance to link the frames that were not on its input or vice versa it links some frames that should not be there at all (at least according to the evaluator), it should not be penalized for it. So only items (frames or arguments) present both on the extracted side and the evaluator's side are taken into consideration. From them generalized metrics of precision, recall and f_1-score are computed.

3.2.7 CzEngVallex evaluation

At the first sight, the idea behind the evaluation using CzEngVallex is quite simple: we need to check, how much the links created by the program correspond to those present in the existing dictionary. Before the actual evaluation, we have to match the extracted frames on both sides with those in the Czech and English parts of CzEngVallex. This task is not simple actually, because frames in

vallex dictionaries include information that cannot be directly derived from the UD annotation, such as functors, i.e. deep-syntactical functions of the valency arguments, whereas UD contains only surface-syntactical dependency relations. So the arguments are matched solely based on the form (cases, prepositions and conjunctions), not the functions. At the same time, CzEngVallex distinguishes obligatory, typical and optional arguments, while the extracted frames do not have such a distinction: they only either take an argument or do not. Apart from the verb lemma, frames must agree in the obligatory arguments and they may agree also in the typical or optional ones, but do not need to. If there is an extracted argument or obligatory vallex argument, that is missing on the other side, the frames cannot be matched.

Since thanks to the typical and optional arguments, it is possible for a frame on whichever side to be matched with multiple frames from the other side, it is necessary to select the best matching. Originally, the algorithm for finding the optimal matching (see 3.1.9), was used here, but it turned out, that the number of matched CzEngVallex frames was too small, nearly all of them were covered. Now, the multiple matches are allowed, so one extracted frame can correspond to various CzEngVallex frames.

3.2.8 Results

In this section the results of the frame linking are finally presented. They are grouped by the linking method, because the main task is to find the most successful linking approach. The following tables describe individual linkers evaluated by the two ways of assessment (on the gold data and in comparison with CzEngVallex) and at the same time separately for frames and arguments. The tables Tab. 3.2 and Tab. 3.3 show results for the word-alignment and the language similarity linker. We can see, that they turn out to be worse than the baseline (which is not surprising at least for the similarity linker, when linking frames between Czech and English), so they should not be part of the final, combined linker.

CS-EN		gold evaluation		CzEngVallex evaluation	
		Base linker	WA linker	Base linker	WA linker
FRAMES	Precision	65.98	71.79	14.45	16.04
	Recall	49.19	27.74	32.07	17.72
	F_1-score	56.3	39.85	19.82	16.61
ARGS	Precision	24.65	37.66	11.27	13.68
	Recall	55.36	37.34	7.43	1.86
	F_1-score	33.94	36.33	7.1	3.17

Table 3.2: Results for the word alignment linker.

On the other side, the linker based on the sentence structure achieves promising results, as the Tab. 3.4 show. Various combinations of the linker's parameters have been proven to be better than the baseline linker and further search of their optimal combination could be a part of future machine learning optimization.

CS-EN		gold evaluation		CzEngVallex evaluation	
		Base linker	WA linker	Base linker	WA linker
FRAMES	Precision	65.98	56.43	14.45	12.94
	Recall	49.19	42.2	32.07	28.44
	F_1-score	53.3	48.24	19.82	17.68
ARGS	Precision	24.41	25.37	11.34	13.59
	Recall	57.38	55.57	7.65	6.18
	F_1-score	36.05	34.71	7.87	7.68

Table 3.3: Results for the language similarity linker.

CS-EN		gold evaluation		CzEngVallex evaluation	
		Base linker	WA linker	Base linker	WA linker
FRAMES	Precision	65.98	75.77	14.45	15.98
	Recall	49.19	55.86	32.07	34.59
	F_1-score	56.3	64.39	19.82	21.66
ARGS	Precision	24.89	25.13	11.34	16.03
	Recall	54.82	52.49	6.51	5.33
	F_1-score	34.11	33.85	7.06	7.19

Table 3.4: Results for the sentence structure linker.

3.3 Experiments with frame linking

One of the applications of the bilingual valency dictionary can be transfer of additional information from one language to another. If there is an existing valency dictionary containing annotation which cannot be obtained from UD data, namely semantic roles, and if we manage to map the valency frames of this dictionary with those extracted ones by this program, then it might be possible to project the additional information to other languages using the bilingual valency frame linking (see Fig. 3.1). This is exactly the case of Vallex, for which we already have implemented and used an algorithm for matching with the extracted frames, so we will perform the projection from Vallex only, but the method would work for any other existing valency dictionary. The projection works regardless of the linking technique. Once the frames have been extracted, it does not matter if the extraction of the valency dictionary has happened via language similarity or using parallel corpus with various possible methods.

The idea is as follows: first we extract and link a bilingual valency dictionary of Czech and another language, then match the extracted Czech frames with Vallex and then we transfer the labels of semantic roles from the Vallex arguments through matches to the extracted Czech arguments and from them further to the extracted arguments of the other language. The languages tried are Slovak and English again. There is no Vallex for Slovak, so it is the ideal candidate for the transfer of semantic roles. However, there are no automatic means to assess the success of such task, we can use only manual annotation. It would be possible to

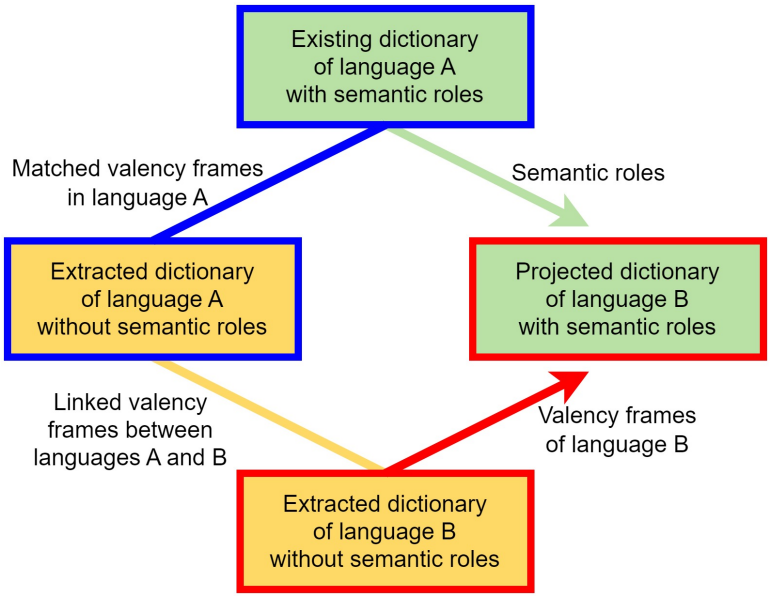


Figure 3.1: Scheme of projection of semantic roles from an existing, non-extracted dictionary into a new language using an extracted bilingual dictionary. Border lines represent language (blue means A, red means B), rectangle fill means presence (green) or absence (yellow) of semantic roles. The semantic roles in the frames of the language A in the existing valency dictionary can be projected via matching and linking into the frames of the language B.

project the Czech functors into English and evaluate this process on EngVallex.[7]I performed projections on a part of Czech-Slovak JRC Acquis. The produced projected "Slovak Vallex" with functors is stored among the other data files.

Another experiment that would be interesting is to try to extract valency frames not from a parallel treebank, but from two separate monolingual treebanks assuming that the two languages are closely related. The basic idea is to extract all verbs from the corpus for each language and then to try to match them according to some similarity metric (see 3.1.6). However, we must realize, that the similarity only helps us to match the verbs. Because the corpora are not parallel, the verb arguments used in the sentences do not correspond to each other. For their linking, it is necessary to use syntactic approach, described in 3.1.5, same as for unrelated languages. This should work slightly worse than with parallel corpus, because we cannot combine the syntactical method with the information from the word alignment. The linking algorithm based on the language similarity can be language-agnostic or may include specific rules for a chosen language pair.

[7]The script performing the projections is called `sk_functor_projector.sh`

Conclusion

Summary of the work

The work aimed to develop a computer program capable of automatic creation of multilingual valency dictionaries. Let us have a closer look on the achievement of the individual goals declared in the introduction according to the assignment of the thesis.

1. Examining the possibilities of multilingual valency frames extraction, including the mapping between corresponding verbs and their arguments.

The possibility of multilingual valency extraction is extensively studied throughout the work. UD annotation relevant for the extraction is analysed in 1.3. The multilingual extraction itself is divided into two parts - monolingual extraction and cross-lingual linking - and various approaches for both parts are analysed in detail. Chapter 2 presents an algorithm for the extraction, which first selects appropriate verbs and then their arguments. Individual linguistic phenomena are treated by separated extraction units, that work on various places of the algorithm (verb selection, argument selection, transformations of selected arguments or postprocessing of the frames after the whole dictionary is built). Some of the units can be parameterized and need to have their parameters tuned. Then in the chapter 3, several methods for mapping the verbs and their valency arguments from multilingual parallel treebank are proposed and analysed. They are based on word-alignment, morphological and syntactic annotation or similarity of words (only for related languages); combination of all these approaches is also considered. All of the methods may have multiple variants, so tuning their parameters is also performed.

2. Distinguishing the approaches that are independent of languages or treebanks from language- or treebank-specific methods.

Both main parts of the work - monolingual extraction and cross-lingual linking - contain language-independent algorithms. The algorithm for monolingual extraction allows extending, language-specific extraction units. Such units are provided for English, Czech and Slovak. The cross-lingual linking contains optional specific extension of the language similarity method (which uses basic string distance metrics) for the language pair Czech-Slovak, which favors particular substitutions that work quite regularly between the two languages. The algorithms does not involve any treatment for a specific treebank. UD annotated treebank is always expected on the input, regardless if it is a part of the set of treebanks UD provides (PUD is used in this work) or it is a non-UD corpus parsed to UD annotation by UDPipe (like JRC Acquis, as used here).

3. Evaluating the quality of the solution on manually annotated data, in particular also the contribution of language-specific approaches compared to the general ones.

The success of monolingual valency frames extraction for English and Czech is evaluated on 100 sentences taken from PUD treebank with manually annotated valency frames (see 2.7). The quality of the cross-lingual linking for Czech-English and Czech-Slovak is assessed in two ways: on the same 100 parallel sentences from PUD (only for Czech-English) and other 100 from Acquis with manually annotated with valency frame links (see **??**), and at the same time by comparison with CzEngVallex, an existing bilingual Czech-English valency dictionary (only for Czech-English; see **??**). Various metrics like precision, recall and f_1-score or simple success rate are used for the evaluation. The extraction and linking of verbs and of arguments are evaluated separately. The contribution of the general, language-independent algorithms is computed by comparison with trivial baseline solutions, whereas the language-specific approaches are evaluated against the general methods as their baselines.

Possible future extensions

There are various possible directions, in which the work could continue. Many decisions based on development testing, like default values of parameters or default exclusion of some extraction units (e.g. frame reduction - see 2.4.6), could be reconsidered when tested on bigger data. Inclusion of oblique dependents into the extracted valency frame can be improved on the level of bilingual linking, with regard to the treatment of the corresponding dependent in the other language, as mentioned in 2.4.5. There is of course a large space for further language-specific extensions, either for the extraction (originally, I planned a separate extension for also for Spanish) or for the linking, at least for other related languages. Various improvements could be done in the output form of the dictionary. In particular, it would seem attractive to me to implement advanced searching system with options of filtering or sorting the results according to various features of the frame pairs. Also better representation of multilingual dictionaries for three or more languages would be appropriate. A major task would be also inclusion of semantic roles into the frames. However, they can be hardly deduced from the UD annotation, so it would require using also other, substantially different kind of data resources. Experiments with projections of semantic roles from Vallex performed in the chapter **??** might serve as a starting point for further efforts.

Appendices

A Installation and requirements

The attachment provided to the thesis includes the whole compressed project. The same project can be downloaded also from the repository `https://github.com/Jankus1994/ud-valency/`. The projects includes the files of the used tools UDPipe, Fast_align and Udapi. Make sure you provide the necessary permissions to the executable files `udpipe` and `fast_align` in the `scripts` directory. Apart from standard Python libraries you will need to install `yattag`, `PyYAML`, `colorama` and `termcolor` (the latter two are used by Udapi).

The `data` directory includes English and Czech PUD files. From JRC Acquis it includes only small parts, since the whole treebank would be too large. You can download the corpus for English, Czech and Slovak and their sentence alignments from `https://wt-public.emm4u.eu/Acquis/JRC-Acquis.3.0/alignmentsHunAlign/index.html`. Make sure you use these alignments from HunAlign; these were used in this work, but the official webpage of JRC Acquis offers also alignments from other sentence aligners.

B Program files and directories

The program uses several tools (see 1.4.6 - 1.4.8) which are not part of it. Although Udapi can be used as an external tool, too, the preferred way to use it is to add user-created blocks into its execution pipeline. These blocks need to be stored in the `block` directory in the UDapi directory structure. The files of the program are thus divided into two parts: files containing code used in UDapi blocks are stored in the `valency` subdirectory of the UDapi `block` directory and the rest is stored in the `scripts` and `data` subdirectories of the main directory `ud-valency`. These two folders together with the Udapi `valency` contain relevant files for the work and are further described below. The whole structure is shown on the Fig. 2.

The `scripts` directory includes multiple shellscripts, most of which can be run to use to program or to reproduce the tests presented in this work and several configuration files. The use of the scripts and configuration files is described in E. The directory contains also important python files, either directly or group in subdirectories, but none of them is meant to by used directly by the user.

The data directory contains subfolders for different types of data files, from the original corpora through various intermediate formats to the final output files for viewing the dictionary. Vallex files and manual annotations for evaluation are also stored in the this directory. The program separates by default each type

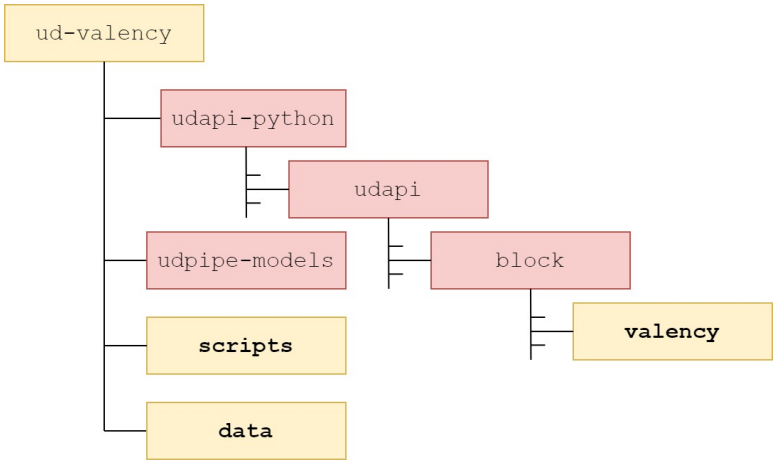


Figure 2: Directory structure of the program. Directories of the used tools are shown in red, custom directories created for this program are shown in yellow.

of file and looks also at its monolingual and bilingual nature (denoted by the letters `m` and `b` in the directory name). The directories can be easily changed in the configuration file (see E.2), but it is not recommended. so the program can use e.g. only one directory for all types of files. A folder often serves as output directory for one part of the program and as input directory for another. There

C Valency dictionary structure

A monolingual valency dictionary is represented as a hash table (python dictionary) with verb lemmas as its keys and verb records as its values. Each verb record has several valency frames defined by the number and types of their arguments. There is a distinction made between the frames and frame arguments as types and as instances. Instances are particular occurrences of the type in the treebank, a type is an abstraction of its instances.

For studying the frames on their own, the two type objects would be sufficient. However, the frame instances (and analogically also argument instances) are needed e.g. for examining frequency of the frame or for displaying example sentences containing the frame. Each frame instance is connected to its frame type and allows showing example sentences for that valency frame. Similarly each frame argument instance is connected to the corresponding frame argument type allowing to highlight the frame arguments in the example sentence and mark their connection to the argument type (see D for more details about viewing the dictionary).

A frame can have multiple (but probably not hundreds) arguments and an argument belongs only to one frame. This holds for both, types and instances. Similarly a type can have multiple instances (even thousands, depending on the treebank), an instance belongs only to one type and this is true for both frames and frame arguments. The scheme of the dictionary structure and is shown on the Fig. 3.

These objects serve for representing a monolingual valency dictionary. Two such dictionaries for different languages can be linked together via connections of the described objects. Corresponding valency frame types will be linked to-

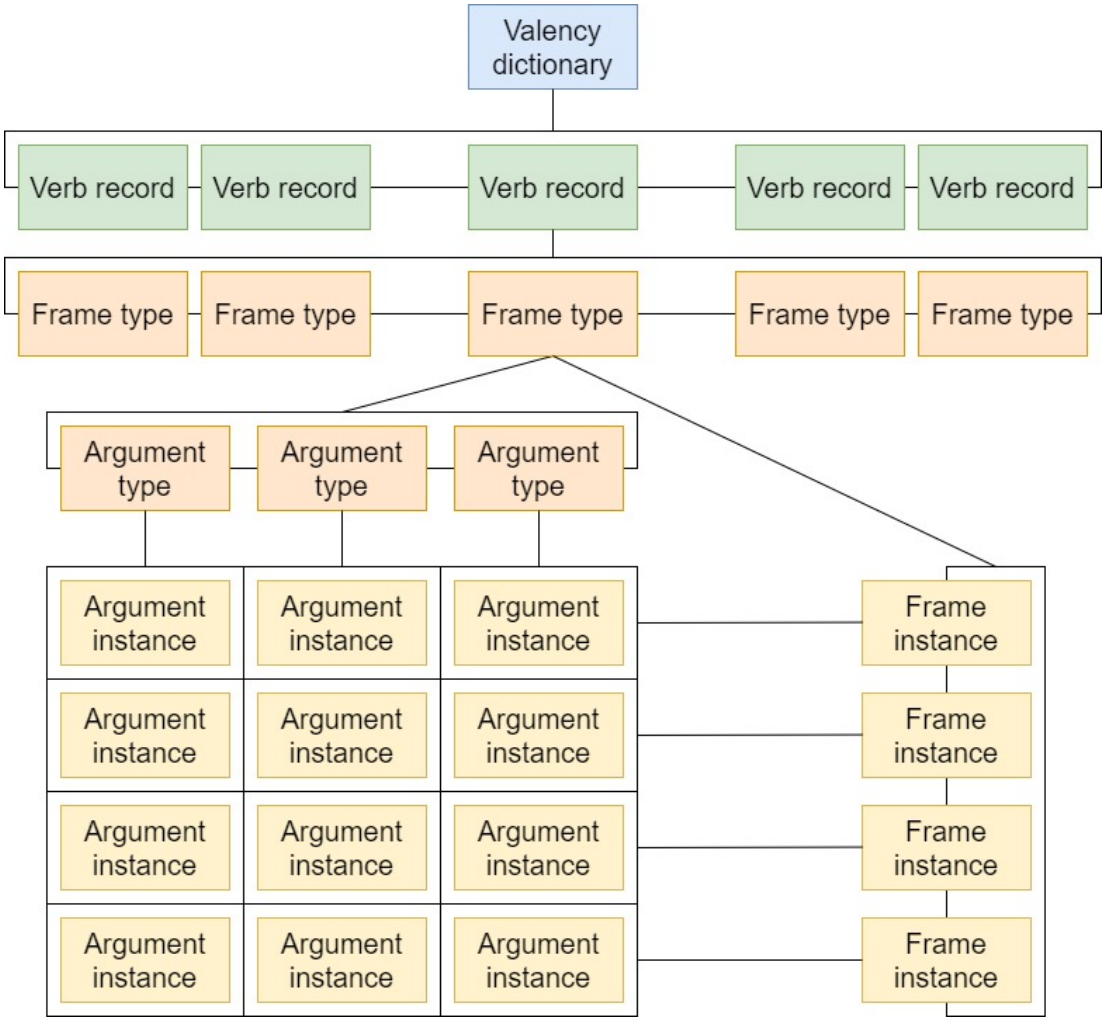


Figure 3: Overview of classes used in valency dictionary structure.

gether and similarly frame instances, argument types and argument instances. If the bilingual valency dictionary is built from a parallel corpus, so the example sentences for both languages correspond to each other, the instance objects can be linked as well. A frame instance can be linked only to one frame instance (still speaking about bilingual dictionary), so each of its argument instances can be linked also at most to one argument instance on the other side. But a frame type can be linked to multiple frame types of the other language, according to its instances, and so do its argument types. The four types of links have also their separate classes.

D Forms of the output

The program presented in this work can be used to produce a monolingual or a bilingual valency dictionary. A multilingual dictionary consisting of three or more languages can be created as a sets of bilingual ones. But their simultaneous visualization is not implemented and remains as a possible extension in the future.

The appropriate form of the output should depend on the expected use of the dictionary. Two main purposes come into consideration: browsing the dictionary by a human user or processing it with a computer for usage in other applications. One option is to come up with a united format for both purposes (probably a text format, similar as in CoNLL-U). However, the dictionary structure, especially for multilingual dictionaries, is quite complex, so it is better to display it in a more comfortable, graphical way, e.g. as a simple HTML page. On the other hand, for some computer applications might be more comfortable to load the dictionary into directly the original structure from a binary file, without need to parse a text file.

In the end, there are three possible output formats of the dictionary (shown with their option in the command line):

- binary format (option **`bin`**)
- text format (option **`text`**)
- HTML format (option **`html`**)

The binary format is meant for further processing of the valency dictionary using its implemented class structure (see C). It is also used as output of the bilingual extraction which is later loaded for the linking (these two tasks are run separately).

Both user outputs (plain text and HTML), are structured according to alphabetically sorted the verb lemmas. The bilingual dictionary is always produced in two files, one for each direction, denoted by their suffix, so e.g. HTML dictionary from a Czech-English treebank (called `treebank`) would consist of files `treebank.cs.html`, sorted according to the Czech lemmas, and `treebank.en.html`, sorted according to the English verbs.

Each verb may have several frames on a lower level. In the bilingual dictionaries, there is an extra level, where the frame is linked with a frame (including its verb) in the other language. The last level in both formats are occurrences of the frame or frame fair in the sentences of the treebank. The levels are denoted

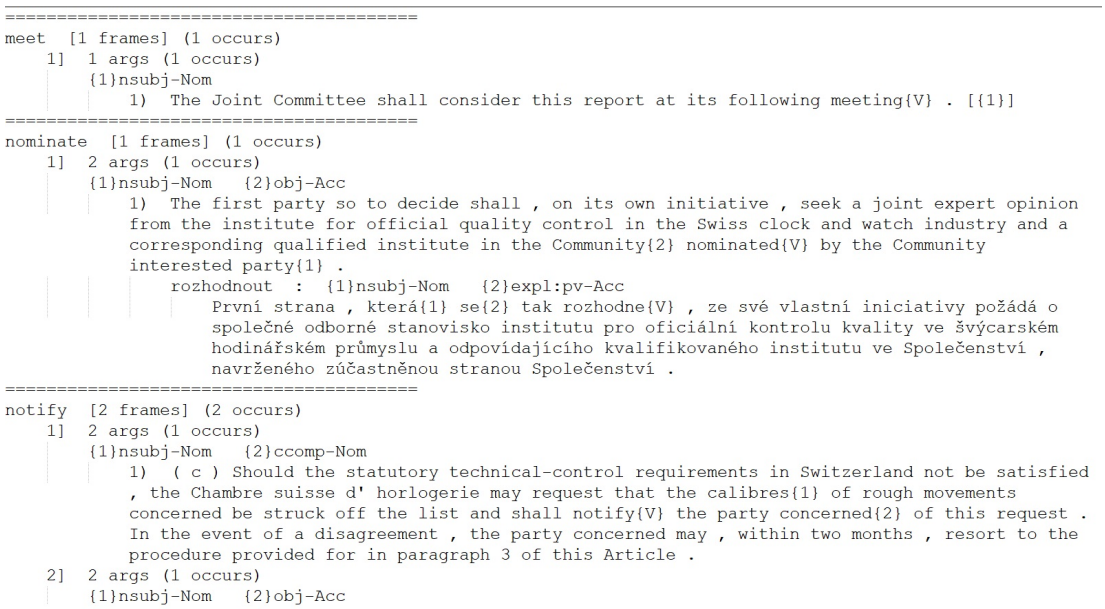


Figure 4: Illustration of the plain text output of the English-Czech valency dictionary.

by indentation in the text format and by hidden lines in the HTML format, that can be shown after clicking the `show` button in the verb line. Different colors of the levels also help understanding the structure of the dictionary. In the text format, the frames of a verb are further indented according to their subframe and superframe structure they form, so the user can see that arguments of a frame form a subset or superset of the another frame's arguments.

The arguments correspondence between arguments of the frame type and their realizations in the example sentences, and also between the linked arguments between the two languages in the case of a bilingual dictionary, is denoted by a number in curly braces (lower-indexed in HTML). In the hypertext format, the verb and arguments in the example sentence are distinguished by different colors. The numbering of the arguments in plain text and HTML format may differ, because the numbers are computed according to different strategies. The ordering of the frames within the verb can also vary.

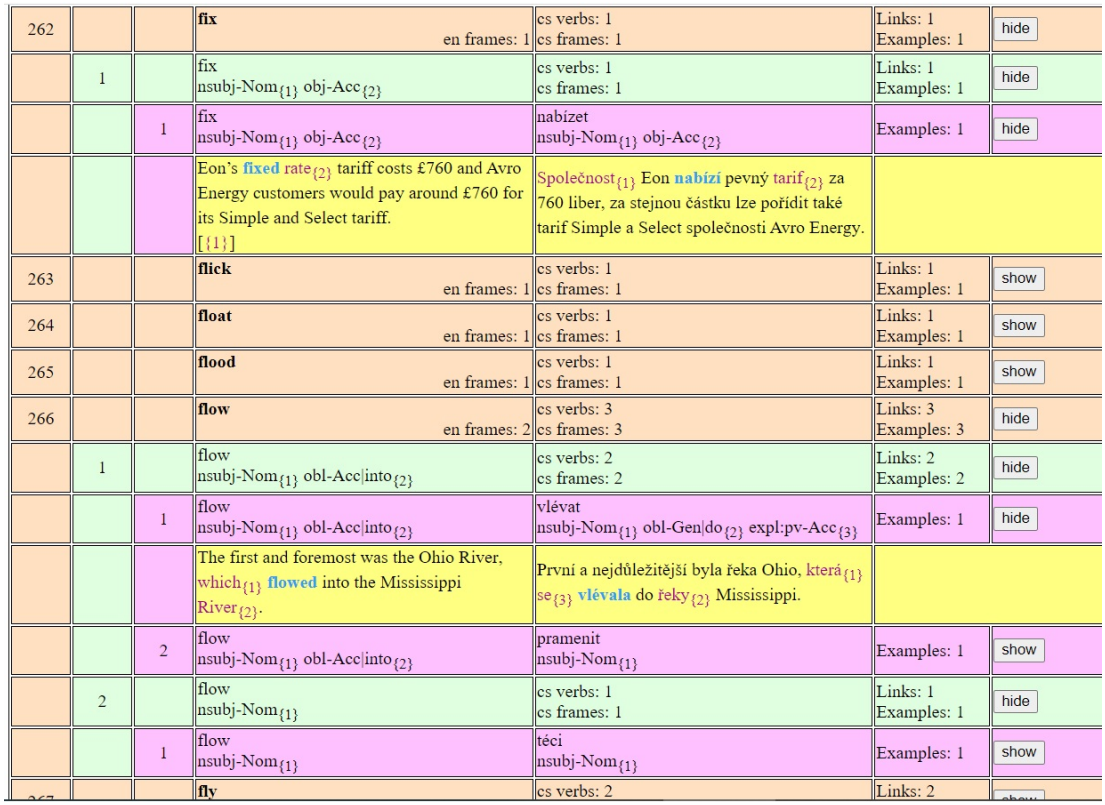
The appearance of the text and HTML output is illustrated on the screenshots in Fig. 4 and Fig. 5.

E Running the program

This section gives detailed instructions to run the program. It describes options for actual use of the application (see E.1), various testing scenarios used in this work (see **??** and configuration files (see E.2).

E.1 Application use

Most of the important code of the program is written in Python, but the necessary tools (UDPipe, fast_align, Udapi) are easier to run from the command line, so the whole program is executed by several shellscripts, that run the Python scripts



262			**fix** en frames: 1	cs verbs: 1 cs frames: 1	Links: 1 Examples: 1	hide
	1		fix nsubj-Nom{1} obj-Acc{2}	cs verbs: 1 cs frames: 1	Links: 1 Examples: 1	hide
		1	fix nsubj-Nom{1} obj-Acc{2}	nabízet nsubj-Nom{1} obj-Acc{2}	Examples: 1	hide
			Eon's **fixed** rate{2} tariff costs £760 and Avro Energy customers would pay around £760 for its Simple and Select tariff. [{1}]	Společnost{1} Eon **nabízí** pevný tarif{2} za 760 liber, za stejnou částku lze pořídit také tarif Simple a Select společnosti Avro Energy.		
263			**flick** en frames: 1	cs verbs: 1 cs frames: 1	Links: 1 Examples: 1	show
264			**float** en frames: 1	cs verbs: 1 cs frames: 1	Links: 1 Examples: 1	show
265			**flood** en frames: 1	cs verbs: 1 cs frames: 1	Links: 1 Examples: 1	show
266			**flow** en frames: 2	cs verbs: 3 cs frames: 3	Links: 3 Examples: 3	hide
	1		flow nsubj-Nom{1} obl-Acc|into{2}	cs verbs: 2 cs frames: 2	Links: 2 Examples: 2	hide
		1	flow nsubj-Nom{1} obl-Acc|into{2}	vlévat nsubj-Nom{1} obl-Gen|do{2} expl:pv-Acc{3}	Examples: 1	hide
			The first and foremost was the Ohio River, which{1} **flowed** into the Mississippi River{2}.	První a nejdůležitější byla řeka Ohio, která{1} se{3} **vlévala** do řeky{2} Mississippi.		
		2	flow nsubj-Nom{1} obl-Acc|into{2}	pramenit nsubj-Nom{1}	Examples: 1	show
	2		flow nsubj-Nom{1}	cs verbs: 1 cs frames: 1	Links: 1 Examples: 1	hide
		1	flow nsubj-Nom{1}	téci nsubj-Nom{1}	Examples: 1	show
			fly	cs verbs: 2	Links: 2	

Figure 5: Illustration of the hypertext output of the English-Czech valency dictionary.

or the tools.

The two main parts of the program - frame extraction and frame linking - are preceded by a sequence of scripts preparing the treebank. These four scripts, stored in the directory `scripts` are used to run the program and are further described below:

- `prep.sh`
- `extr.sh`
- `link.sh`
- `output.sh`

They are all run with the `bash` command with several parameters. The first one is a language code (for monolingual extraction; e.g. `cs` for Czech and `en` for English) or two language codes connected with a hyphen (for bilingual extraction or linking; e.g. `cs-en` for the Czech-English pair). The codes distinguish the sentences in the parallel CoNLL-U file and are used to denote a variant of two corresponding files, either completely monolingual (separate versions of a bilingual treebank stored in two file) or differing in the relation to the languages (e.g. two one-directional word alignments or the final dictionaries, which are always sorted according to one of the languages). They also decide the use of implicit, language-specific extractor, if it exists (the use of another extractor needs to be specified in the configuration file). If nothing language-specific should be used,

the codes can be rather arbitrary, but they must be the same during the whole process.

The second parameter of all the scripts except the preparing one is the desired form of the output. This can be `bin`, `text` or `html` (see D). The last parameter of all the scripts (i.e. the second one for `prep.sh` and the third one for the rest) defines the name of the treebank. This parameter is optional; the treebank name can be specified also in the configuration file, from where it is taken, if it is missing in the command line. All other options, can be set only via configuration files.

The script `prep.sh` implements a pipeline of actions preparing the corpus. In the monolingual use, it only runs UDPipe for tokenizing, tagging and parsing. If it is used for preparation of a parallel bilingual treebank, it performs also word-alignment by Fast_align between the tokenization and tagging. It also uses two minor custom scripts to merge one-directional alignments into two-directional (`fa_interpreter.py`) and to merge two monolingual CoNLL-U files into a parallel, bilingual one, with sentences distinguished by the language codes (`conllu_merger.py`).

The corpora should on the input of `prep.sh` should be int text format, but sentence-segmented, with each sentence on a separate line. They must be store in the `sents` directory (or whatever it is redefined with in the configuration file). Two files forming a parallel bilingual corpus must be sentence-aligned, i.e. have the same number of lines. Original forms of non-UD corpora are meant to be stored in the `corpora` folder, their conversion into the sentence format is left upon the user, with the exception of Acquis, used in this work . UD treebanks for the monolingual extraction can be used directly as the input if `extr.sh`, but for the bilingual use, they need to be at least merged and propably also word-aligned (if a linker using word-alignment will be used).

The script `extr.sh` runs Udapi with extraction blocks. The language codes are relevant for implicit choice of language-specific extractors, if they exist. The script `link.sh` runs the frame linking and also may use language-specific linkers. For producing a bilingual valency dictionary, the extraction part needs to produce a binary output, so that it can be later loaded by the linking script. In order to avoid repeating extractions and linkings, the script `output.sh` loads already extracted (monolingual use) or also linked (bilingual use) binary data and transforms them into one of the other output formats (plain text or HTML).

E.2 Configuration files

Configuration files allow various more advanced settings of the program, than parameters on the command line. They are all in `yaml` format, which is comfortable for the human user and easily parsable into a Python dictionary with the respective library. They are stored in a subdirectory called `configs`, inside the `scripts` folder. There is one main configuration file (`config.yaml`) and two more specific ones describing properties of individual extractors (`extr_config.yaml`) and linkers (`link_config.yaml`). The possible values and the original value (in case the used rewrites it) are always stated in the comment in the configuration file. The values of some of the configuration keys may remain unspecified (this is always explicitly said in the comments, too).

The main configuration file `config.yaml` specifies the treebank name (but it is used only if omitted in the command line), UDPipe models, explicit extractors

(used independently of the specified language codes), names of the other two special configuration files, and the names of all directories used by the program for various purposes.

The `extr_config.yaml` specifies which extraction units of various extending extractors (Main, English, Czech and Slovak - see 2.4 - 2.6 for the description of the units) should be used. This is specified by characters `0` (excluded) or `1` (included). Only unit parameters may have different values. Some units are deactivated by default (this is stated for them in the file), the majority is activated. The units may have parameters. In that case, there are more values for the given key: the first one is the actual value describing the inclusion or exclusion of the unit, the rest are the parameters.

Parameters of frame linkers (see them in are specified in the `link_config.yaml`. It has a two-level structure, which loads into nested dictionaries. Most of the values is binary, marked by `0` or `1`, some, like tresholds are numerical (this is always specified for such parameter in the configuration file).

Bibliography

Bojar, Ondřej (2002). "Automatická extrakce lexikálně-syntaktických údajů z korpusu". Master Thesis. Prague: Institute of Formal, Applied Linguistics, Faculty of Mathematics, and Physics, Charles University

Buchholz, Sabine and Erwin Marsi (2006). "CoNLL-X shared task on Multilingual Dependency Parsing". In: *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X)*, pp. 149–164. URL: https://aclanthology.org/W06-2920

Cinková, Silvie et al. (2014). *EngVallex - English Valency Lexicon*. URL: http://hdl.handle.net/11858/00-097C-0000-0023-4337-2

— (2021). *EngVallex - English Valency Lexicon 2.0*. URL: http://hdl.handle.net/11234/1-3526

Croft, W. Bruce et al. (2017). "Linguistic Typology meets Universal Dependencies". In: *International Workshop on Treebanks and Linguistic Theories*. URL: https://ceur-ws.org/Vol-1779/05croft.pdf

de Marneffe, Marie-Catherine, Timothy Dozat, et al. (May 2014). "Universal Stanford dependencies: A cross-linguistic typology". In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), pp. 4585–4592. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/1062_Paper.pdf

de Marneffe, Marie-Catherine, Christopher D. Manning, et al. (2021). "Universal Dependencies". In: *Computational Linguistics* 47.2, pp. 255–308. URL: https://aclanthology.org/2021.cl-2.11

Droganova, Kira and Daniel Zeman (2019). "Towards Deep Universal Dependencies". In: *Proceedings of the Fifth International Conference on Dependency Linguistics (Depling, SyntaxFest 2019)*, pp. 144–152. URL: https://aclanthology.org/W19-7717

Dryer, Matthew S. and Martin Haspelmath, eds. (2013). *WALS Online (v2020.3)*. Zenodo. DOI: 10.5281/zenodo.7385533. URL: https://doi.org/10.5281/zenodo.7385533

Dyer, Chris, Victor Chahuneau, and Noah A. Smith (2013). "A Simple, Fast, and Effective Reparameterization of IBM Model 2". In: *Proceedings of NAACL-HLT 2013*, pp. 644–648. URL: https://aclanthology.org/N13-1073

Fillmore, Charles J. (1968a). *The Case for case*

— (1968b). *Types of Lexical Information*

Gerdes, Kim et al. (2019). "Improving Surface-syntactic Universal Dependencies (SUD): MWEs and deep syntactic features". In: *Proceedings of the 18th In-*

ternational Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2019), pp. 126–132. URL: https://aclanthology.org/W19-7814

Gruzitis, Normunds, Gunta Nespore-Berzkalne, and Baiba Saulite (2018). "Creation of Latvian FrameNet based on Universal Dependencies". In: *Proceedings of the LREC 2018 Workshop International FrameNet Workshop 2018: Multilingual Framenets and Constructicons*. URL: http://lrec-conf.org/workshops/lrec2018/W5/pdf/9_W5.pdf

Jindal, Ishan et al. (2022). "Universal Proposition Bank 2.0". In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pp. 1700–1711. URL: https://aclanthology.org/2022.lrec-1.181

Lopatková, Markéta, Václava Kettnerová, Eduard Bejček, et al. (2016). *VALLEX 3.0*. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University. URL: http://hdl.handle.net/11234/1-2307

Lopatková, Markéta, Václava Kettnerová, Jiří Mírovský, et al. (2022). *VALLEX 4.5*. Prague. URL: http://hdl.handle.net/11234/1-4756

Nivre, Joakim et al. (2020). "Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection". In: *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 4034–4043. URL: https://aclanthology.org/2020.lrec-1.497

Petrov, Slav et al. (2012). "A universal part-of-speech tagset". In: *Proceedings of LREC*

Popel, Martin, Zdeněk Žabokrtský, and Martin Vojtek (2017). "Udapi: Universal API for Universal Dependencies". In: *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pp. 96–101. URL: http://universaldependencies.org/udw17/pdf/UDW12.pdf

Przepiórkowski, Adam and Agnieszka Patejuk (2018). "Arguments and Adjuncts in Universal Dependencies". In: *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 3837–3852. URL: https://aclanthology.org/C18-1324

Sarkar, Anoop and Daniel Zeman (2000). "Automatic Extraction of Subcategorization Frames for Czech". In: *COLING 2000 Volume 2: The 18th International Conference on Computational Linguistics*. URL: https://aclanthology.org/C00-2100

Schuster, Sebastian and Christopher D. Manning (2016). "Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 2371–2378. URL: https://aclanthology.org/L16-1376

Shi, Tianze and Lillian Lee (2018). "Valency-Augmented Dependency Parsing". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1277–1291. URL: https://aclanthology.org/D18-1159

Steinberger, Ralf et al. (2006). "The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages". In: *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*. URL: https://joint-research-centre.ec.europa.eu/language-technology-resources/jrc-acquis_en#acknowledgement--reference-publication

Straka, Milan and Jana Straková (2017). "Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe". In: *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*

— (2019). *Universal Dependencies 2.5 Models for UDPipe (2019-12-06)*. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. URL: http://hdl.handle.net/11234/1-3131

Tesnière, Lucien (1959). *Éléments de syntaxe structurale*

UD web (2023). URL: https://universaldependencies.org/ (visited on 2023)

Urešová, Zdeňka, Eva Fučíková, Jan Hajič, et al. (2018). "Defining Verbal Synonyms: between Syntax and Semantics". In: *Proceedings of the 17th International Workshop on Treebanks and Linguistic Theories (TLT 2018)*, pp. 75–90. URL: http://ufal.mff.cuni.cz/biblio/attachments/2018-uresova-m2250687249202608710.pdf

Urešová, Zdeňka, Eva Fučíková, Eva Hajičová, et al. (2019). "Parallel Dependency Treebank Annotated with Interlinked Verbal Synonym Classes and Roles". In: *Proceedings of the 18th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2019)*, pp. 38–50. URL: https://aclanthology.org/W19-7805

Urešová, Zdeňka, Eva Fučíková, and Jana Šindlerová (2016). "CzEngVallex: a bilingual Czech-English valency lexicon". In: *The Prague Bulletin of Mathematical Linguistics*, pp. 17–50

Urešová, Zdeňka, Karolina Zaczynska, et al. (2022). "Making a Semantic Event-type Ontology Multilingual". In: *Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022)*, pp. 1332–1334

Vernerová, Anna (2019). "Lexicographic treatment of the valency aspects of verbal diatheses". PhD thesis

Žabokrtský, Zdeněk (2005). *Valency Lexicon of Czech Verbs (PhD thesis)*. Praha, Czechia: ÚFAL MFF UK

Zeman, Daniel (2008). "Reusable Tagset Conversion Using Tagset Drivers". In: *Proceedings of LREC*. URL: http://lrec-conf.org/proceedings/lrec2008/pdf/66_paper.pdf