Coordination Structures in Dependency Treebanks

Martin Popel, David Mareček, Jan Štěpánek, Daniel Zeman, Zdeněk Žabokrtský
Charles University in Prague, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics (ÚFAL)
Malostranské náměstí 25, CZ-11800 Praha, Czechia
{popel|marecek|stepanek|zeman|zabokrtsky}@ufal.mff.cuni.cz

Abstract

Paratactic syntactic structures are notoriously difficult to represent in dependency formalisms. This has painful consequences such as high frequency of parsing errors related to coordination. In other words, coordination is a pending problem in dependency analysis of natural languages. This paper tries to shed some light on this area by bringing a systematizing view of various formal means developed for encoding coordination structures. We introduce a novel taxonomy of such approaches and apply it to treebanks across a typologically diverse range of 26 languages. In addition, empirical observations on convertibility between selected styles of representations are shown too.

1 Introduction

In the last decade, dependency parsing has gradually been receiving visible attention. One of the reasons is the increased availability of dependency treebanks, be they results of genuine dependency annotation projects or converted automatically from previously existing phrase-structure treebanks.

In both cases, a number of decisions have to be made during the construction or conversion of a dependency treebank. The traditional notion of dependency does not always provide unambiguous solutions, e.g. when it comes to attaching functional words. Worse, dependency representation is at a loss when it comes to representing paratactic linguistic phenomena such as coordination, whose nature is symmetric (two or more conjuncts play the same role), as opposed to the head-modifier asymmetry of dependencies.[1]

The dominating solution in treebank design is to introduce artificial rules for the encoding of coordination structures within dependency trees using the same means that express dependencies, i.e., by using edges and by labeling of nodes or edges. Obviously, any tree-shaped representation of a coordination structure (CS) must be perceived only as a "shortcut" since relations present in coordination structures form an undirected cycle, as illustrated already by Tesnière (1959). For example, if a noun is modified by two coordinated adjectives, there is a (symmetric) coordination relation between the two conjuncts and two (asymmetric) dependency relations between the conjuncts and the noun.

However, as there is no obvious linguistic intuition telling us which tree-shaped CS encoding is better and since the degree of freedom has several dimensions, one can find a number of distinct conventions introduced in particular dependency treebanks. Variations exist both in topology (tree shape) and labeling. The main goal of this paper is to give a systematic survey of the solutions adopted in these treebanks.

Naturally, the interplay of dependency and coordination links in a single tree leads to serious parsing issues.[2] The present study does not try to decide which coordination style is the best from the parsing point of view.[3] However, we believe that our survey will substantially facilitate experiments in this direction in the future, at least by exploring and describing the space of possible candidates.

[1] We use the term *modifier* (or *child*) for all types of dependent nodes including *arguments*.

[2] CSs have been reported to be one of the most frequent sources of parsing errors (Green and Žabokrtský, 2012; McDonald and Nivre, 2007; Kübler et al., 2009; Collins, 2003). Their impact on quality of dependency-based machine translation can also be substantial; as documented on an English-to-Czech dependency-based translation system (Popel and Žabokrtský, 2009), 39% of serious translation errors which are caused by wrong parsing have to do with coordination.

[3] There might be no such answer, as different CS conventions might serve best for different applications or for different parser architectures.

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 517–527,
Sofia, Bulgaria, August 4-9 2013.

The rest of the paper is structured as follows. Section 2 describes some known problems related to CS. Section 3 shows possible "styles" for representing CS. Section 4 lists treebanks whose CS conventions we studied. Section 5 presents empirical observations on CS convertibility. Section 6 concludes the paper.

2 Related work

Let us first recall the basic well-known characteristics of CSs.

In the simplest case of a CS, a coordinating conjunction joins two (usually syntactically and semantically compatible) words or phrases called conjuncts. Even this simplest case is difficult to represent within a dependency tree because, in the words of Lombardo and Lesmo (1998): *Dependency paradigms exhibit obvious difficulties with coordination because, differently from most linguistic structures, it is not possible to characterize the coordination construct with a general schema involving a head and some modifiers of it.*
Proper formal representation of CSs is further complicated by the following facts:

- CSs with more than two conjuncts (multi-conjunct CSs) exist and are frequent.
- Besides "private" modifiers of individual conjuncts, there are modifiers shared by all conjuncts, such as in *"Mary came and cried"*. Shared modifiers may appear alongside with private modifiers of particular conjuncts.
- Shared modifiers can be coordinated, too: *"big and cheap apples and oranges"*.
- Nested (embedded) coordinations are possible: *"John and Mary or Sam and Lisa"*.
- Punctuation (commas, semicolons, three dots) is frequently used in CSs, mostly with multi-conjunct coordinations or juxtapositions which can be interpreted as CSs without conjunctions (e.g. *"Don't worry, be happy!"*).
- In many languages, comma or other punctuation mark may play the role of the main coordinating conjunction.
- The coordinating conjunction may be a multiword expression (*"as well as"*).
- Deficient CSs with a single conjunct exist.
- Abbreviations like *"etc."* comprise both the conjunction and the last conjunct.
- Coordination may form very intricate structures when combined with ellipsis. For example, a conjunct can be elided while its arguments remain in the sentence, such as in the following traditional example: *"I gave the books to Mary and the records to Sue."*
- The border between paratactic and hypotactic surface means of expressing coordination relations is fuzzy. Some languages can use enclitics instead of conjunctions/prepositions, e.g. Latin *"Senatus Populus*que *Romanus"*. Purely hypotactic surface means such as the preposition in *"John with Mary"* occur too.[4]
- Careful semantic analysis of CSs discloses additional complications: if a node is modified by a CS, it might happen that it is the node itself (and not its modifiers) what should be semantically considered as a conjunct. Note the difference between *"red and white wine"* (which is synonymous to *"red wine and white wine"*) and *"red and white flag of Poland"*. Similarly, *"five dogs and cats"* has a different meaning than *"five dogs and five cats"*.

Some of these issues were recognized already by Tesnière (1959). In his solution, conjuncts are connected by vertical edges directly to the head and by horizontal edges to the conjunction (which constitutes a cycle in every CS). Many different models have been proposed since, out of which the following are the most frequently used ones:

- MS = Mel'čuk style used in the Meaning-Text Theory (MTT): the first conjunct is the head of the CS, with the second conjunct attached as a dependent of the first one, third conjunct under the second one, etc. Coordinating conjunction is attached under the penultimate conjunct, and the last conjunct is attached under the conjunction (Mel'čuk, 1988),
- PS = Prague Dependency Treebank (PDT) style: all conjuncts are attached under the coordinating conjunction (along with shared modifiers, which are distinguished by a special attribute) (Hajič et al., 2006),

[4] As discussed by Stassen (2000), all languages seem to have some strategy for expressing coordination. Some of them lack the paratactic surface means (the so called WITH-languages), but the hypotactic surface means are present almost always.

- SS = Stanford parser style:[5] the first conjunct is the head and the remaining conjuncts (as well as conjunctions) are attached under it.

One can find various arguments supporting the particular choices. MTT possesses a complex set of linguistic criteria for identifying the governor of a relation (see Mazziotta (2011) for an overview), which lead to MS. MS is preferred in a rule-based dependency parsing system of Lombardo and Lesmo (1998). PS is advocated by Štěpánek (2006) who claims that it can represent shared modifiers using a single additional binary attribute, while MS would require a more complex co-indexing attribute. An argumentation of Tratz and Hovy (2011) follows a similar direction: *We would like to change our [MS] handling of coordinating conjunctions to treat the coordinating conjunction as the head [PS] because this has fewer ambiguities than [MS]...*

We conclude that the influence of the choice of coordination style is a well-known problem in dependency syntax. Nevertheless, published works usually focus only on a narrow ad-hoc selection of few coordination styles, without giving any systematic perspective.

Choosing a file format presents a different problem. Despite various efforts to standardize linguistic annotation,[6] no commonly accepted standard exists. The primitive format used for CoNLL shared tasks is widely used in dependency parsing, but its weaknesses have already been pointed out (cf. Straňák and Štěpánek (2010)). Moreover, particular treebanks vary in their contents even more than in their format, i.e. each treebank has its own way of representing prepositions or different granularity of syntactic labels.

3 Variations in representing coordination structures

Our analysis of variations in representing coordination structures is based on observations from a set of dependency treebanks for 26 languages.[7] In accordance with the usual conventions, we assume that each sentence is represented by one dependency tree, in which each node corresponds to one token (word or punctuation mark). Apart from that, we deliberately limit ourselves to CS representations that have shapes of connected subgraphs of dependency trees.

We limit our inventory of means of expressing CSs within dependency trees to (i) tree topology (presence or absence of a directed edge between two nodes, Section 3.1), and (ii) node labeling (additional attributes stored insided nodes, Section 3.2).[8] Further, we expect that the set of possible variations can be structured along several dimensions, each of which corresponds to a certain simple characteristic (such as choosing the leftmost conjunct as the CS head, or attaching shared modifiers below the nearest conjunct). Even if it does not make sense to create the full Cartesian product of all dimensions because some values cannot be combined, it allows to explore the space of possible CS styles systematically.[9]

3.1 Topological variations

We distinguish the following dimensions of topological variations of CS styles (see Figure 1):

Family – configuration of conjuncts. We divide the topological variations into three main groups, labeled as Prague (fP), Moscow (fM), and

[5] We use the already established MS-PS-SS distinction to facilitate literature overview; as shown in Section 3, the space of possible coordination styles is much richer.

[6] For example, TEI (TEI Consortium, 2013), PML (Hana and Štěpánek, 2012), SynAF (ISO 24615, 2010).

[7] The primary data sources are the following: *Ancient Greek*: Ancient Greek Dependency Treebank (Bamman and Crane, 2011), *Arabic*: Prague Arabic Dependency Treebank 1.0 (Smrž et al., 2008), *Basque*: Basque Dependency Treebank (larger version than CoNLL 2007 generously provided by IXA Group) (Aduriz and others, 2003), *Bulgarian*: BulTreeBank (Simov and Osenova, 2005), *Czech*: Prague Dependency Treebank 2.0 (Hajič et al., 2006), *Danish*: Danish Dependency Treebank (Kromann et al., 2004), *Dutch*: Alpino Treebank (van der Beek and others, 2002), *English*: Penn TreeBank 3 (Marcus et al., 1993), *Finnish*: Turku Dependency Treebank (Haverinen et al., 2010), *German*: Tiger Treebank (Brants et al., 2002), *Greek (modern)*: Greek Dependency Treebank (Prokopidis et al., 2005), *Hindi, Bengali and Telugu*: Hyderabad Dependency Treebank (Husain et al., 2010), *Hungarian*: Szeged Treebank (Csendes et al., 2005), *Italian*: Italian Syntactic-Semantic Treebank (Montemagni and others, 2003), *Latin*: Latin Dependency Treebank (Bamman and Crane, 2011), *Persian*: Persian Dependency Treebank (Rasooli et al., 2011), *Portuguese*: Floresta sintá(c)tica (Afonso et al., 2002), *Romanian*: Romanian Dependency Treebank (Călăcean, 2008), *Russian*: Syntagrus (Boguslavsky et al., 2000), *Slovene*: Slovene Dependency Treebank (Džeroski et al., 2006), *Spanish*: AnCora (Taulé et al., 2008), *Swedish*: Talbanken05 (Nilsson et al., 2005), *Tamil*: TamilTB (Ramasamy and Žabokrtský, 2012), *Turkish*: METU-Sabanci Turkish Treebank (Atalay et al., 2003).

[8] Edge labeling can be trivially converted to node labeling in tree structures.

[9] The full Cartesian product of variants in Figure 1 would result in topological 216 variants, but only 126 are applicable (the inapplicable combinations are marked with "—" in Figure 1). Those 126 topological variants can be further combined with labeling variants defined in Section 3.2.

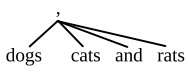
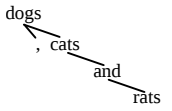
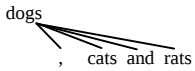
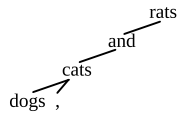
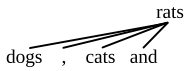
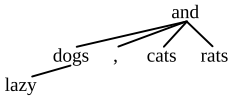
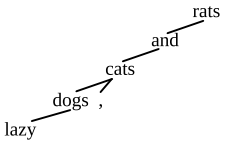
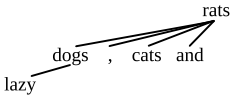
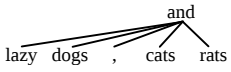
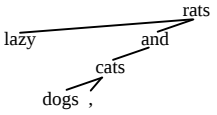
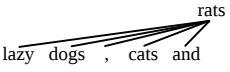
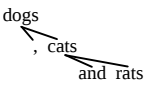
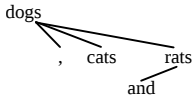
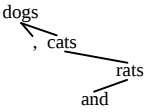
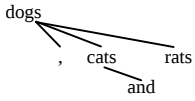
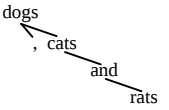
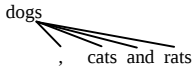
Main family	Prague family (code fP) [14 treebanks]	Moscow family (code fM) [5 treebanks]	Stanford family (code fS) [6 treebanks]
Choice of head			
Head on left (code hL) [10 treebanks]			
Head on right (code hR) [14 treebanks]			
Mixed head (code hM) [1 treebank]	A mixture of hL and hR		
Attachment of shared modifiers			
Shared modifier below the nearest conjunct (code sN) [15 treebanks]			
Shared modifier below head (code sH) [11 treebanks]			
Attachment of coordinating conjunction			
Coordinating conjunction below previous conjunct (code cP) [2 treebanks]	—		
Coordinating conjunction below following conjunct (code cF) [1 treebank]	—		
Coordinating conjunction between two conjuncts (code cB) [8 treebanks]	—		
Coordinating conjunction as the head (code cH) is the only applicable style for the Prague family [14 treebanks]		—	—
Placement of punctuation			
values pP [7 treebanks], pF [1 treebank] and pB [15 treebanks] are analogous to cP, cF and cB (but applicable also to the Prague family)			

Figure 1: Different coordination styles, variations in tree topology. Example phrase: *"(lazy) dogs, cats and rats"*. Style codes are described in Section 3.1.

Stanford (fS) families.[10] This first dimension distinguishes the configuration of conjuncts: in the Prague family, all the conjuncts are siblings governed by one of the conjunctions (or a punctuation fulfilling its role); in the Moscow family, the conjuncts form a chain where each node in the chain depends on the previous (or following) node; in the Stanford family, the conjuncts are siblings except for the first (or last) conjunct, which is the head.[11]

Choice of head – leftmost or rightmost. In the Prague family, the head can be either the leftmost[12] (hL) or the rightmost (hR) conjunction or punctuation. Similarly, in the Moscow and Stanford families, the head can be either the leftmost (hL) or the rightmost (hR) conjunct. A third op-

[10] Names are chosen purely as a mnemonic device, so that Prague Dependency Treebank belongs to the Prague family, Mel'čuk style belongs to the Moscow family, and Stanford parser style belongs to the Stanford family.

[11] Note that for CSs with just two conjuncts, fM and fS may look exactly the same (depending on the attachment of conjunctions and punctuation as described below).

[12] For simplicity, we use the terms left and right even if their meaning is reversed for languages with right-to-left writing systems such as Arabic or Persian.

tion (hM) is to mix hL and hR based on some criterion, e.g. the Persian treebank uses hR for coordination of verbs and hL otherwise. For the experiments in Section 5, we choose the head which is closer to the parent of the whole CS, with the motivation to make the edge between CS head and its parent shorter, which may improve parser training.

Attachment of shared modifiers. Shared modifiers may appear before the first conjunct or after the last one. Therefore, it seems reasonable to attach shared modifiers either to the CS head (sH), or to the nearest (i.e. first or last) conjunct (sN).

Attachment of coordinating conjunctions. In the Moscow family, conjunctions may be either part of the chain of conjuncts (cB), or they may be put outside of the chain and attached to the previous (cP) or following (cF) conjunct. In the Stanford family, conjunctions may be either attached to the CS head (and therefore *between* conjuncts) (cB), or they may be attached to the previous (cP) or the following (cF) conjunct. The cB option in both Moscow and Stanford families, treats conjunctions in the same way as conjuncts (with respect to topology only). In the Prague family, there is just one option available (cH) – one of the conjunctions is the CS head while the others are attached to it.

Attachment of punctuation. Punctuation tokens separating conjuncts (commas, semicolons etc.) could be treated the same way as conjunctions. However, in most treebanks it is treated differently, so we consider it as well. The values pP, pF and pB are analogous to cP, cF and cB except that punctuation may be also attached to the conjunction in case of pP and pF (otherwise, a comma before the conjunction would be non-projectively attached to the member following the conjunction).

The three established styles mentioned in Section 2 can be defined in terms of the newly introduced abbreviations: PS = fPhRsHcHpB, MS = fMhLsNcBp?, and SS = fShLsNcBp?.[13]

3.2 Labeling variations

Most state-of-the-art dependency parsers can produce labeled edges. However, the parsers produce only one label per edge. To fully capture CSs, we need more than one label, because there are several aspects involved (see the initial assumptions in Section 3): We need to identify the coordinating conjunction (its POS tag might not be enough), conjuncts, shared modifiers, and punctuation that separates conjuncts. Besides that, there should be a label classifying the dependency relation between the CS and its parent.

Some of the information can be retrieved from the topology of the tree and the "main label" of each node, but not everything. The additional information can be attached to the main label, but such approach obscures the logical structure.

In the **Prague family**, there are two possible ways to label a conjunction and conjuncts:

Code dU ("**d**ependency labeled at the **u**pper level of the CS"). The dependency relation of the whole CS to its parent is represented by the label of the conjunction, while the conjuncts are marked with a special label for conjuncts (e.g. `ccof` in the Hyderabad Dependency Treebank).

Code dL ("**l**ower level"). The CS is represented by a coordinating conjunction (or punctuation if there is no conjunction) with a special label (e.g. `Coord` in PDT). Subsequently, each conjunct has its own label that reflects the dependency relation towards the parent of the whole CS, therefore, conjuncts of the same CS can have different labels, e.g. *"Who[SUBJ] and why[ADV] did it?"*

Most Prague family treebanks use sH, i.e. shared modifiers are attached to the head (coordinating conjunction). Each child of the head has to belong to one of three sets: conjuncts, shared modifiers, and punctuation or additional conjunctions. In PDT, conjuncts, punctuation and additional conjunctions are recognized by specific labels. Any other children of the head are shared modifiers.

In the **Stanford and Moscow families**, one of the conjuncts is the head. In practice, it is never labeled as a conjunct explicitly, because the fact that it is a conjunct can be deduced from the presence of conjuncts among its children. Usually, the other conjuncts are labeled as conjuncts; conjunctions and punctuation also have a special label. This type of labeling corresponds to the dU type.

Alternatively (as found in the Turkish treebank, dL), all conjuncts in the Moscow chain have their own dependency labels and the fact that they are conjuncts follows from the `COORDINATION` labels of the conjunction and punctuation nodes between them.

To represent shared modifiers in the Stan-

[13] The question marks indicate that the original Mel'čuk and Stanford parser styles ignore punctuation.

ford and Moscow families, an additional label is needed again to distinguish between private and shared modifiers since they cannot be distinguished topologically. Moreover, if nested CSs are allowed, a binary label is not sufficient (i.e. "shared" versus "private") because it also has to indicate which conjuncts the shared modifier belongs to.[14]

We use the following binary flag codes for capturing which CS participants are distinguished in the annotation: m01 = shared modifiers annotated; m10 = conjuncts annotated; m11 = both annotated; m00 = neither annotated.

4 Coordination Structures in Treebanks

In this section, we identify the CS styles defined in the previous section as used in the primary treebank data sources; statistical observations (such as the amount of annotated shared modifiers) presented here, as well as experiments on CS-style convertibility presented in Section 5.2, are based on the normalized shapes of the treebanks as contained in the HamleDT 1.0 treebank collection (Zeman et al., 2012).[15]

Some of the treebanks were downloaded individually from the web, but most of them came from previously published collections for dependency parsing campaigns: six languages from CoNLL-2006 (Buchholz and Marsi, 2006), seven languages from CoNLL-2007 (Nivre et al., 2007), two languages from CoNLL-2009 (Hajič and others, 2009), three languages from ICON-2010 (Husain et al., 2010). Obviously, there is a certain risk that the CS-related information contained in the source treebanks was slightly biased by the properties of the CoNLL format upon conversion. In addition, many of the treebanks were natively dependency-based (cf. the 2nd column of Table 1), but some were originally based on constituents and thus specific converters to the CoNLL format had to be created (for instance, the Spanish phrase-structure trees were converted to dependencies using a procedure described by Civit et al. (2006); similarly, treebank-specific converters have been used for other languages). Again, there is some risk that the CS-related information contained in treebanks resulting from such conversions is slightly different from what was intended in the very primary annotation.

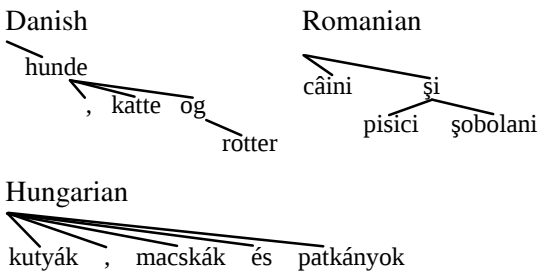


Figure 2: Annotation styles of a few treebanks do not fit well into the multidimensional space defined in Section 3.1.

There are several other languages (e.g. Estonian or Chinese) which are not included in our study, despite of the fact that constituency treebanks do exist for them. The reason is that the choice of their CS style would be biased, because no independent converters exist – we would have to convert them to dependencies ourselves. We also know about several more dependency treebanks that we have not processed yet.

Table 1 shows 26 languages whose treebanks we have studied from the viewpoint of their CS styles. It gives the basic quantitative properties of the treebanks, their CS style in terms of the taxonomy introduced in Section 3, as well as statistics related to CSs: the average number of CSs per 100 tokens, the average number of conjuncts per one CS, the average number of shared modifiers per one CS,[16] and the percentage of nested CSs among all CSs. The reader can return to Figure 1 to see the basic statistics on the "popularity" of individual design decisions among the developers of dependency treebanks or constituency treebank converters.

CS styles of most treebanks are easily classifiable using the codes introduced in Section 3, plus a few additional codes:

- p0 = punctuation was removed from the treebank.

[14] This is not needed in Prague family where shared modifiers are attached to the conjunction provided that each shared modifier is shared by conjuncts that form a full subtree together with their coordinating conjunctions; no exceptions were found during the annotation process of the PDT.

[15] A subset of the treebanks whose license terms permit redistribution is available directly at http://ufal.mff.cuni.cz/hamledt/.

[16] All non-Prague family treebanks are marked sN and m00 or m10, (i.e. shared modifiers not marked in the original annotation, but attached to the head conjunct) because we found no counterexamples (modifiers attached to a conjunct, but not the nearest one). The HamleDT normalization procedure contains a few heuristics to detect shared modifiers, but it cannot recover the missing distinction reliably, so the numbers in the "SMs/CJ" column are mostly underestimated.

Language	Orig. type	Data set	Sents.	Tokens	Original CS style code	CSs / 100 tok.	CJs / CS	SMs / CS	Nested CS[%]	RT UAS
Ancient Greek	dep	prim.	31 316	461 782	fP hR sH cH pB dL m11	6.54	2.17	0.16	10.3	97.86
Arabic	dep	C07	3 043	116 793	fP hL sH cH pB dL m00	3.76	2.42	0.13	10.6	96.69
Basque	dep	prim.	11 225	151 593	fP hR sN cH pP dU m00	3.37	2.09	0.03	5.1	99.32
Bengali	dep	I10	1 129	7 252	fP hR sH cH pP dU m11	4.87	1.71	0.05	24.1	99.97
Bulgarian	phr	C06	13 221	196 151	fS hL sN cB pB dU m10	2.99	2.19	0.00	0.0	99.74
Czech	dep	C07	25 650	437 020	fP hR sH cH pB dL m11	4.09	2.16	0.20	14.6	99.42
Danish	dep	C06	5 512	100 238	fS* hL sN cP pB dU m10	3.68	1.93	0.13	7.5	99.76
Dutch	phr	C06	13 735	200 654	fP hR sN cH pP dU m10	2.06	2.17	0.05	3.3	99.47
English	phr	C07	40 613	991 535	fP hR sH cH pB dU m10	2.07	2.33	0.05	6.3	99.84
Finnish	dep	prim.	4 307	58 576	fS hL sN cB pB dU m10	4.06	2.41	0.00	6.4	99.70
German	phr	C09	38 020	680 710	fM hL sN cP pP dU m10	2.79	2.09	0.01	0.0	99.73
Greek	dep	C07	2 902	70 223	fP hR sH cH pB dL m11	3.25	2.48	0.18	7.2	99.43
Hindi	dep	I10	3 515	77 068	fP hR sH cH pP dU m11	2.45	1.97	0.04	10.3	98.35
Hungarian	phr	C07	6 424	139 143	fT hX sN cX pX dL m00	2.37	1.90	0.01	2.2	99.84
Italian	dep	C07	3 359	76 295	fS hL sN cB pB dU m10	3.32	2.02	0.03	3.8	99.51
Latin	dep	prim.	3 473	53 143	fP hR sH cH pB dL m11	6.74	2.24	0.41	12.3	97.45
Persian	dep	prim.	12 455	189 572	fM*hM sN cB pP dU m00	4.18	2.10	0.18	3.7	99.82
Portuguese	phr	C06	9 359	212 545	fS hL sN cB pB dU m10	2.51	1.95	0.26	11.1	99.16
Romanian	dep	prim.	4 042	36 150	fP* hR sN cH p0 dU m10	1.80	2.00	0.00	0.0	100.00
Russian	dep	prim.	34 895	497 465	fM hL sN cB p0 dU m10	4.02	2.02	0.07	3.9	99.86
Slovene	dep	C06	1 936	35 140	fP hR sH cH pB dL m00	4.31	2.49	0.00	10.8	98.87
Spanish	phr	C09	15 984	477 810	fS hL sN cB pB dU m10	2.79	1.98	0.14	12.7	99.24
Swedish	phr	C06	11 431	197 123	fM hL sN cF pF dU m10	3.94	2.19	0.13	0.7	99.66
Tamil	dep	prim.	600	9 581	fP hR sH cH pB dL m11	1.66	2.46	0.22	3.8	99.67
Telugu	dep	I10	1 450	5 722	fP hR sH cH pP dU m11	3.48	1.59	0.06	5.0	100.00
Turkish	dep	C07	5 935	69 695	fM hR sN cB pB dL m10	3.81	2.04	0.00	34.3	99.23

Table 1: Overview of analyzed treebanks. prim. = primary source; C06–C09 = CoNLL 2006–2009; I10 = ICON 2010; SM = shared modifier; CJ = conjunct; Nested CS = portion of CSs participating in nested CSs (both as the inner and outer CS); RT UAS = unlabeled attachment score of the roundtrip experiment described in Section 5. Style codes are defined in Sections 3 and 4.

- fM* = Persian treebank uses a mix of fM and fS: fS for coordination of verbs and fM otherwise.

Figure 2 shows three other anomalies:

- fS* = Danish treebank employs a mixture of fS and fM, where the last conjunct is attached indirectly via the conjunction.
- fP* = Romanian treebank omits punctuation tokens and multi-conjunct coordinations get split.
- fT = Hungarian Szeged treebank uses "Tesnière family" – disconnected graphs for CSs where conjuncts (and conjunction and punctuation) are attached directly to the parent of CS, and so the other style dimensions are not applicable (hX, cX, pX).

5 Empirical Observations on Convertibility of Coordination Styles

The various styles cannot represent the CS-related information to the same extent. For example, it is not possible to represent nested CSs in the Moscow and Stanford families without significantly changing the number of possible labels.[17] The dL style (which is most easily applicable to the Prague family) can represent coordination of different dependency relations. This is again not possible in the other styles without adding e.g. a special "prefix" denoting the relations.

We can see that the Prague family has a greater expressive power than the other two families: it can represent complex CSs using just one additional binary label, distinguishing between shared modifiers and conjuncts. A similar additional label is needed in the other styles to distinguish between shared and private modifiers.

Because of the different expressive power, converting a CS from one style to another may lead to a loss of information. For example, as

[17] Mel'čuk uses "grouping" to nest CSs – cf. related solutions involving coindexing or bubble trees (Kahane, 1997). However, these approaches were not used in any of the researched treebanks. To combine grouping with shared modifiers, each group in a tree should have a different identifier.

there is no way of representing shared modifiers in the Moscow family without an additional attribute, converting a CS with shared modifiers from Prague to Moscow family makes the modifiers private. When converting back, one can use certain heuristics to handle the most obvious cases, but sometimes the modifiers will stay private (very often, the nature of a modifier depends on context or is debatable even for humans, e.g. *"Young boys and girls"*).

5.1 Transformation algorithm

We developed an algorithm to transform one CS style to another. Two subtasks must be solved by the algorithm: identification of individual CSs and their participants, and transforming of the individual CSs.

Obviously, the individual CSs cannot be transformed independently because of coordination nesting. For instance, when transforming a nested coordination from the Prague style to the Moscow style (e.g. to fMhL), the leftmost conjunct in the inner (lower) coordination must climb up to become the head of the inner CS, but then it must climb up once again to become the head of the outer (upper) CS too. This shows that inner CSs must be transformed first.

We tackle this problem by a depth-first recursion. When going down the tree, we only recognize all the participants of the CSs, classify them and gather them in a separate data structure (one for each visited CS). The following four types of CS participants are distinguished: coordinating conjunctions, conjuncts, shared modifiers, and punctuations that separate conjuncts.[18] No change of the tree is performed during these descent steps.

When returning back from the recursion (i.e., when climbing from a node back up to its parent), we test whether the abandoned node is the topmost node of some CS. If so, then this CS is transformed, which means that its participants are rehanged and relabelled according the the target CS style.

This procedure naturally guarantees that the inner CSs are transformed first and that all CSs are transformed when the recursions returns to the root.

5.2 Roundtrip experiment

The number of possible conversion directions obviously grows quadratically with the number of styles. So far, we limited ourselves only to conversions from/to the style of the HamleDT treebank collection, which contains all the treebanks under our study already converted into a common scheme. The common scheme is based on the conventions of PDT, whose CS style is fPhRsHcHpB.[19]

We selected nine styles (3 families times 3 head choices) and transformed all the HamleDT scheme treebanks to these nine styles and back, which we call a *roundtrip*. Resulting averaged unlabeled attachment scores (UAS, evaluated against the HamleDT scheme) in the last column of Table 1 indicate that the percentage of transformation errors (i.e. tokens attached to a different parent after the roundtrip) is lower than 1% for 20 out of the 26 languages.[20] A manual inspection revealed two main error sources. First, as noted above, the Stanford and Moscow families have lower expressive power than the Prague family, so naturally, the inverse transformation was ambiguous and the transformation heuristics were not capable of identifying the correct variant every time. Second, we also encountered inconsistencies in the original treebanks (which we were not trying to fix in HamleDT for now).

6 Conclusions and Future Work

We described a (theoretically very large) space of possible representations of CSs within the dependency framework. We pointed out a range of details that make CSs a really complex phenomenon; anyone dealing with CSs in treebanking should take these observations into account.

We proposed a taxonomy of those approaches

[18] Conjuncts are explicitly marked in most styles. Coordinating conjunctions can be usually identified with the help of dependency labels and POS tags. Punctuation separating conjuncts can be detected with high accuracy using simple rules. If shared modifiers are not annotated (code m00 or m10), one can imagine rule-based heuristics or special classifiers trained to distinguish shared modifiers. For the experiments in this section, we use the HamleDT gold annotation attribute `is_shared_modifier`.

[19] As documented in Zeman et al. (2012), the normalization procedures used in HamleDT embrace many other phenomena as well (not only those related to coordination), and involve both structural transformation and dependency relation relabeling.

[20] Table 1 shows that Latin and Ancient Greek treebanks have on average more than 6 CSs per 100 tokens, more than 2 conjuncts per CS, and Latin has also the highest number of shared modifiers per CS. Therefore the percentage of nodes affected by the roundtrip is the highest for these languages and the lower roundtrip UAS is not surprising.

that have been argued for in literature or employed in real treebanks.

We studied 26 existing treebanks of different languages. For each value of each dimension in Figure 1, we found at least one treebank where the value is used; even so, several treebanks take their own unique path that cannot be clearly classified under the taxonomy (the taxonomy could indeed be extended, for the price of being less clearly arranged).

We discussed the convertibility between the various styles and implemented a universal tool that transforms between any two styles of the taxonomy. The tool achieves a roundtrip accuracy close to 100%. This is important because it opens the door to easily switching coordination styles for parsing experiments, phrase-to-dependency conversion etc.

While the focus of this paper is to explore and describe the expressive power of various annotation styles, we did not address the learnability of the styles by parsers. That will be a complementary point of view, and thus a natural direction of future work for us.

Acknowledgments

We thank the providers of the primary data resources. The work on this project was supported by the Czech Science Foundation grants no. P406/11/1499 and P406/2010/0875, and by research resources of the Charles University in Prague (PRVOUK). This work has been using language resources developed and/or stored and/or distributed by the LINDAT-Clarin project of the Ministry of Education of the Czech Republic (project LM2010013). Further, we would like to thank Jan Hajič, Ondřej Dušek and four anonymous reviewers for many useful comments on the manuscript of this paper.

References

Itzair Aduriz et al. 2003. Construction of a Basque dependency treebank. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories*.

Susana Afonso, Eckhard Bick, Renato Haber, and Diana Santos. 2002. "Floresta sintá(c)tica": a treebank for Portuguese. In *LREC*, pages 1968–1703.

Nart B. Atalay, Kemal Oflazer, and Bilge Say. 2003. The annotation process in the Turkish treebank. In *Proceedings of the 4th Intern. Workshop on Linguistically Interpreteted Corpora (LINC)*.

David Bamman and Gregory Crane. 2011. The Ancient Greek and Latin dependency treebanks. In *Language Technology for Cultural Heritage*, Theory and Applications of Natural Language Processing, pages 79–98. Springer Berlin Heidelberg.

Igor Boguslavsky, Svetlana Grigorieva, Nikolai Grigoriev, Leonid Kreidlin, and Nadezhda Frid. 2000. Dependency treebank for Russian: Concept, tools, types of information. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*, pages 987–991. Association for Computational Linguistics Morristown, NJ, USA.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of CoNLL*, pages 149–164.

Montserrat Civit, Maria Antònia Martí, and Núria Bufí. 2006. Cat3LB and Cast3LB: From constituents to dependencies. In *FinTAL*, volume 4139 of *Lecture Notes in Computer Science*, pages 141–152. Springer.

Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational linguistics*, 29(4):589–637.

Dóra Csendes, János Csirik, Tibor Gyimóthy, and András Kocsor. 2005. The Szeged treebank. In *TSD*, volume 3658 of *Lecture Notes in Computer Science*, pages 123–131. Springer.

Mihaela Călăcean. 2008. Data-driven dependency parsing for Romanian. Master's thesis, Uppsala University, August.

Sašo Džeroski, Tomaž Erjavec, Nina Ledinek, Petr Pajas, Zdeněk Žabokrtský, and Andreja Žele. 2006. Towards a Slovene dependency treebank. In *LREC 2006*, pages 1388–1391, Genova, Italy. European Language Resources Association (ELRA).

Nathan Green and Zdeněk Žabokrtský. 2012. Hybrid combination of constituency and dependency trees into an ensemble dependency parser. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pages 19–26, Avignon, France. Association for Computational Linguistics.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, Zdeněk Žabokrtský, and Magda Ševčíková-Razímová. 2006. Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.

Jan Hajič et al. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Jirka Hana and Jan Štěpánek. 2012. Prague markup language framework. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 12–21, Stroudsburg, PA, USA. Association for Computational Linguistics, Association for Computational Linguistics.

Katri Haverinen, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Filip Ginter, and Tapio Salakoski. 2010. Treebanking Finnish. In *Proceedings of the Ninth International Workshop on Treebanks and Linguistic Theories (TLT9)*, pages 79–90.

Samar Husain, Prashanth Mannem, Bharat Ambati, and Phani Gadde. 2010. The ICON-2010 tools contest on Indian language dependency parsing. In *Proceedings of ICON-2010 Tools Contest on Indian Language Dependency Parsing*, Kharagpur, India.

ISO 24615. 2010. Language resource management – Syntactic annotation framework (SynAF).

Sylvain Kahane. 1997. Bubble trees and syntactic representations. In *Proceedings of the 5th Meeting of the Mathematics of the Language, DFKI, Saarbrucken*.

Matthias T. Kromann, Line Mikkelsen, and Stine Kern Lynge. 2004. Danish dependency treebank.

Sandra Kübler, Erhard Hinrichs, Wolfgang Maier, and Eva Klett. 2009. Parsing coordinations. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 406–414, Athens, Greece, March. Association for Computational Linguistics.

Vincenzo Lombardo and Leonardo Lesmo. 1998. Unit coordination and gapping in dependency theory. In *Processing of Dependency-Based Grammars; proceedings of the workshop. COLING-ACL*, Montreal.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330.

Nicolar Mazziotta. 2011. Coordination of verbal dependents in Old French: Coordination as a specified juxtaposition or apposition. In *Proceedings of International Conference on Dependency Linguistics (DepLing 2011)*.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131.

Igor A. Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press.

Simonetta Montemagni et al. 2003. Building the Italian syntactic-semantic treebank. In *Building and using Parsed Corpora*, Language and Speech series, pages 189–210, Dordrecht. Kluwer.

Jens Nilsson, Johan Hall, and Joakim Nivre. 2005. MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity. In *Proceedings of the NODALIDA Special Session on Treebanks*.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL 2007 Shared Task. EMNLP-CoNLL*, June.

Martin Popel and Zdeněk Žabokrtský. 2009. Improving English-Czech Tectogrammatical MT. *The Prague Bulletin of Mathematical Linguistics*, (92):1–20.

Prokopis Prokopidis, Elina Desipri, Maria Koutsombogera, Harris Papageorgiou, and Stelios Piperidis. 2005. Theoretical and practical issues in the construction of a Greek dependency treebank. In *Proceedings of the 4th Workshop on Treebanks and Linguistic Theories (TLT)*, pages 149–160.

Loganathan Ramasamy and Zdeněk Žabokrtský. 2012. Prague dependency style treebank for Tamil. In *Proceedings of LREC 2012*, pages 23–25, İstanbul, Turkey. European Language Resources Association.

Mohammad Sadegh Rasooli, Amirsaeid Moloodi, Manouchehr Kouhestani, and Behrouz Minaei-Bidgoli. 2011. A syntactic valency lexicon for Persian verbs: The first steps towards Persian dependency treebank. In *5th Language & Technology Conference (LTC): Human Language Technologies as a Challenge for Computer Science and Linguistics*, pages 227–231, Poznań, Poland.

Kiril Simov and Petya Osenova. 2005. Extending the annotation of BulTreeBank: Phase 2. In *The Fourth Workshop on Treebanks and Linguistic Theories (TLT 2005)*, pages 173–184, Barcelona, December.

Otakar Smrž, Viktor Bielický, Iveta Kouřilová, Jakub Kráčmar, Jan Hajič, and Petr Zemánek. 2008. Prague Arabic dependency treebank: A word on the million words. In *Proceedings of the Workshop on Arabic and Local Languages (LREC) 2008*, pages 16–23, Marrakech, Morocco. European Language Resources Association.

Leon Stassen. 2000. And-languages and with-languages. *Linguistic Typology*, 4(1):1–54.

Jan Štěpánek. 2006. *Capturing a Sentence Structure by a Dependency Relation in an Annotated Syntactical Corpus (Tools Guaranteeing Data Consistence)* (in Czech). Ph.D. thesis, Charles Univer-

sity in Prague, Faculty of Mathematics and Physics, Prague, Czech Republic.

Pavel Straňák and Jan Štěpánek. 2010. Representing layered and structured data in the CoNLL-ST format. In Alex Fang, Nancy Ide, and Jonathan Webster, editors, *Proceedings of the Second International Conference on Global Interoperability for Language Resources*, pages 143–152, Hong Kong, China. City University of Hong Kong, City University of Hong Kong.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel annotated corpora for Catalan and Spanish. In *LREC*. European Language Resources Association.

TEI Consortium. 2013. TEI P5: Guidelines for Electronic Text Encoding and Interchange.

Lucien Tesnière. 1959. *Eléments de syntaxe structurale*. Paris.

Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of EMNLP*, pages 1257–1268, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.

Leonoor van der Beek et al. 2002. Chapter 5. The Alpino dependency treebank. In *Algorithms for Linguistic Processing NWO PIONIER Progress Report*, Groningen, The Netherlands.

Daniel Zeman, David Mareček, Martin Popel, Loganathan Ramasamy, Jan Štěpánek, Zdeněk Žabokrtský, and Jan Hajič. 2012. HamleDT: To parse or not to parse? In *Proceedings of LREC 2012*, pages 2735–2741, İstanbul, Turkey. European Language Resources Association.