Prague Arabic Dependency Treebank: A Word on the Million Words

Otakar Smrž Viktor Bielický Iveta Kouřilová Jakub Kráčmar Jan Hajič Petr Zemánek

Institute of Formal and Applied Linguistics, Charles University in Prague
Malostranské náměstí 25, Prague 1, 118 00, Czech Republic
`http://ufal.mff.cuni.cz/padt/`   `<padt@ufal.mff.cuni.cz>`

Abstract

Prague Arabic Dependency Treebank (PADT) consists of refined multi-level linguistic annotations over the language of Modern Written Arabic. The kind of morphological and syntactic information comprised in PADT differs considerably from that of the Penn Arabic Treebank (PATB). This paper overviews the character of PADT and its motivations, and reports on converting and enhancing the PATB data in order to be included into PADT. The merged, rule-checked and revised annotations, which amount to over one million words, as well as the open-source computational tools developed in the project are considered for publication this year.

1. Introduction

Prague Arabic Dependency Treebank (PADT) provides refined linguistic annotations inspired by the Functional Generative Description theory (Sgall et al., 1986; Hajičová and Sgall, 2003) and the Prague Dependency Treebank project (Hajič et al., 2006). The multi-level description scheme discerns functional morphology, analytical dependency syntax, and tectogrammatical representation of linguistic meaning. PADT is maintained by the Institute of Formal and Applied Linguistics, Charles University in Prague. The initial version of PADT (Hajič et al., 2004a) covered over one hundred thousand words of text.

PADT was included in the CoNLL 2006 and CoNLL 2007 Shared Task on dependency parsing (Nivre et al., 2007) or in other parsing experiments (Corston-Oliver et al., 2006). The morphological data and methodology of PADT were also used for training automatic taggers (Hajič et al., 2005). PADT is discussed in detail in (Žabokrtský and Smrž, 2003; Hajič et al., 2004b; Smrž, 2007b; Smrž and Hajič, 2008).

Penn Arabic Treebank (PATB) is the largest such resource for Modern Written Arabic that is annotated with structural morphological features, morph-oriented English glosses, and labelled phrase-structure syntactic trees in the predicate-argument style of the Penn Treebank (Marcus et al., 1993). PATB was developed at the Linguistic Data Consortium, University of Pennsylvania, and was published gradually in four major releases (Maamouri et al., 2004a, 2005a,b,c). The source texts are distributed also independently as part of the Arabic Gigaword (Graff, 2007).

PATB has been used mostly for the availability of morphological annotations and fully vocalized word forms. Processing the data with machine-learning techniques has resulted in a number of morphological taggers (Habash and Rambow, 2005; Smith et al., 2005; Hajič et al., 2005) and diacritizers (Nelken and Shieber, 2005; Zitouni et al., 2006; Habash and Rambow, 2007). The syntactic information was exploited in particular for parsing (Kulick et al., 2006), grammar extraction (Habash and Rambow, 2004), and automatic case assignment (Habash et al., 2007).

The PATB treebank is further described in (Maamouri and Bies, 2004; Buckwalter, 2004a; Maamouri et al., 2004b).

This paper explores the possibility to merge both of these treebanks into a uniform resource that would exceed the existing ones in the level of linguistic detail, accuracy, and quantity. While we advance the PADT style of annotations in this effort, we also largely benefit from the amount of disambiguated information available in PATB.

The new more than one-million-word treebank denoted as PADT 2.0 combines original Prague annotations with the transformed and enhanced Penn data. The preliminary contents of these components are enumerated in Table 1.

2. Functional Morphology

Due to the impact of PATB, the computational linguistics community is well aware of the Buckwalter Arabic Morphological Analyzer (Buckwalter, 2002, 2004b). This system can be characterized as following the lexical–incremental approach to morphology (Stump, 2001), implying that the only clue for discovering a word's morphosyntactic properties is through its explicit morphs and their supposed prototypical functions.

The functional view of language pursued in PADT requires, on the contrary, an inferential–realizational morphological model capable of more appropriate and deeper generalizations. ElixirFM (Smrž, 2007a,b) is the novel implementation that replaces any earlier functional approximations used in PADT, which were developed also thanks to the Buckwalter analyzer (Smrž and Pajas, 2004).

In order to illustrate the differences in the morphological description of PATB versus PADT, let us discuss a few examples. One disambiguated word in the PATB data might offer this information:

Form	`All~Asilokiy~apu`	اللاّسِلْكِيَّةُ
Morph	`Al` + `lAsilokiy~` + `ap` + `u`	
Tag	`DET+ADJ+NSUFF_FEM_SG+CASE_DEF_NOM`	
Gloss	the + wireless / radio + [fem.sg.] + [def.nom.]	
Lemma	[`lAsilokiy~_1`]	لَاسِلْكِيّ
Root	*implicit in the lexicon*	

The entry spells out the full inflected word form in the Buckwalter transliteration, identifies its structure, and describes it with tags and glosses. It also provides the citation form of the lexeme that the word form represents. Other information relevant to the lexeme, like its derivational root, might be stored implicitly in the Buckwalter lexicon, but is not readily available in the treebank.

	Data Set	Corpus 'words'	Functional Morphology tokens	 paras	 docs	Dependency Syntax tokens	 paras	 docs	Tectogrammatics tokens	 paras	 docs
Prague	**AEP**	99360	**116717**	3006	327	**116717**	3006	327	**9690**	242	29
	EAT	48371	**55097**	1667	207	**55097**	1667	207	**13934**	436	58
	ASB	11881	**14254**	558	36	**14254**	558	36			
	NHR	21445	**25329**	426	34	**12613**	209	17			
	HYT	85683	**100537**	1782	204	**41855**	796	91	**5228**	106	10
	XIN	61500	**71548**	2389	321	**41716**	1429	196	**2042**	75	13
Penn	**1v3**	141515	**161217**	4790	628	**161217**	4790	628			
	2v2	140821	**163973**	2929	476	**163973**	2929	476			
	3v2	335250	**394466**	12445	589	**394466**	12445	589			
	4v1	149784	**178720**	5618	361						
Prague		328240	**383482**	9828	1129	**282252**	7665	874	**30894**	859	110
Penn		767370	**898376**	25782	2054	**719656**	20164	1693			
PADT 2.0		1095610	**1281858**	35610	3183	**1001908**	27829	2567	**30894**	859	110

Table 1: Expected contents of PADT 2.0. The Prague data sets AEP and EAT cover parts of the Arabic English Parallel News (Ma, 2004) and the full English-Arabic Treebank (Bies, 2006), while ASB, NHR, HYT, and XIN are selected from the Arabic Gigaword (Graff, 2007). The Penn data correspond to the parts and versions of PATB, modulo duplicate documents.

The corresponding PADT entry by ElixirFM would yield:

<table>
<tr><td>Form</td><td>al-lA-silkIyaTu</td><td><i>al-lā-silkīyatu</i> اَللَّاسِلْكِيَّةُ</td></tr>
<tr><td>Morph</td><td>al >| lA >| FiCL |< Iy |< aT |<< "u"</td><td></td></tr>
<tr><td>Tag</td><td>A-----FS1D</td><td></td></tr>
</table>

<table>
<tr><td>Form</td><td>lA-silkIy</td><td><i>lā-silkīy</i> لَاسِلْكِيّ</td></tr>
<tr><td>Morph</td><td>lA >| FiCL |< Iy</td><td></td></tr>
<tr><td>Root</td><td>"s l k"</td><td></td></tr>
<tr><td>Reflex</td><td>wireless, radio</td><td></td></tr>
<tr><td>Class</td><td>adjective</td><td></td></tr>
</table>

The interpretation is a little more formal. The lexeme of the given grammatical class and meaning is inflected in the parameters expressed by the tag—the adjective is inflected for feminine gender, singular number, nominative case, and definite state. Both the inflected word form and the citation form are explicit in their morphological structure—they are specified via the underlying template of morphs and the inherited root. Merging the template with the root produces the form in the ArabTEX notation, from which the orthographic string or its phonetic version can be generated.

In this very instance, both of the treebank entries seem more or less equivalent. With some other kinds of words, however, the PATB morphology systematically fails to determine many of the contextual and lexical parameters:

<table>
<tr><td>Form</td><td>waOuxoraY</td><td>وَأُخْرَى</td></tr>
<tr><td>Morph</td><td>wa + OuxoraY</td><td></td></tr>
<tr><td>Tag</td><td>CONJ+ADJ</td><td></td></tr>
<tr><td>Gloss</td><td>and + other / another / additional</td><td></td></tr>
</table>

<table>
<tr><td>Lemma</td><td>[OuxoraY_1]</td><td>أُخْرَى</td></tr>
</table>

The word to analyze is in fact two lexical words, 'and' and 'other', joined in writing. There are two morphs and two tags, one for the conjunction, one for the adjective. There is yet only one explicit lemma. There are no details about the gender, number, case, or state of the adjective. Linguistically, though, the adjective is feminine singular with some possible, but not actual, ambiguity in case and state, and the lexeme's citation form is not as indicated.

The complete analysis in PADT would rather supply these individual tokens:

<table>
<tr><td>Form</td><td>'u_hrY</td><td><i>ʾuḫrā</i> أُخْرَى</td><td>wa</td><td><i>wa</i> وَ</td></tr>
<tr><td>Morph</td><td>FuCLY |<< "u"</td><td></td><td>"wa"</td><td></td></tr>
<tr><td>Tag</td><td>A-----FS1I</td><td></td><td>C---------</td><td></td></tr>
</table>

<table>
<tr><td>Form</td><td>'A_har</td><td><i>ʾāḫar</i> آخَر</td><td>wa</td><td><i>wa</i> وَ</td></tr>
<tr><td>Morph</td><td>HACaL</td><td></td><td>"wa"</td><td></td></tr>
<tr><td>Root</td><td>"' _h r"</td><td></td><td>"w"</td><td></td></tr>
<tr><td>Reflex</td><td>other, another</td><td></td><td>and</td><td></td></tr>
<tr><td>Class</td><td>adjective</td><td></td><td>conjunction</td><td></td></tr>
</table>

ElixirFM carefully designs the morphophonemic patterns of the templates, as well as the phonological rules hidden in the >| or |<< operators. This greatly simplifies the morphological rules proper, both inflectional and derivational. Inspired by functional programming in Haskell (Forsberg and Ranta, 2004), ElixirFM implements many generalizations of classical grammars (Fischer, 2002), and suggest even some new abstractions (Smrž, 2007b; compare the approach to patterns in Ryding, 2005; Yaghi and Yagi, 2004).

One would expect that the most underspecified words in a treebank might be those with weak morphology (Habash et al., 2007). In our final example, though, the problem lies elsewhere—how do we motivate the morphs, how do we define the tokens, how do we interpret the tags, and how do we ensure the uniformity of this information in all the data?

PATB does not separate the future marker *sa-* from an indicative verb, but does handle distinct tokens if the markers are the stand-alone *sawfa* or *lan*. Likewise, perhaps unintentionally, *li-* is not tokenized if followed by a jussive, but it is tokenized if followed by a subjunctive. There is an easy fix to this redundancy if you notice, but why run the risk of singleton particle–verb forms, tags, tokens, etc.?

```
|> "s l k" <| [
    FaCaL                  `verb`  [ "proceed", "behave" ]
        `imperf`  FCuL,
    FiCL                   `noun`  [ "wire", "thread" ]
        `plural`  HaFCAL,
    FiCL |< Iy             `adj`   [ "wire", "by wire" ],
    lA >| FiCL |< Iy       `adj`   [ "wireless", "radio" ],
    FuCUL                  `noun`  [ "behavior", "conduct" ],
    FuCUL |< Iy            `adj`   [ "behavioral" ],
    MaFCaL                 `noun`  [ "road", "method" ]
        `plural`  MaFACiL                                  ]
```

proceed, behave	I(*u*) *salak*	سَلَك
wire, thread	(*ʾaslāk* أَسلَاك) *silk*	سِلك
wire, by wire	*silkīy*	سِلكِيّ
wireless, radio	*lā-silkīy*	لَاسِلكِيّ
behavior, conduct	*sulūk*	سُلُوك
behavioral	*sulūkīy*	سُلُوكِيّ
road, method	*maslak*	مَسلَك
	(*masāliku*	مَسَالِكُ)

Figure 1: Excerpt from the ElixirFM lexicon of the entries nested under the *s l k* سلك root, and a layout generated from it.

Form	`sayad~aEiy`	سَيَدَّعِي
Morph	`sa` + `ya` + `d~aEiy` + `(null)`	
Tag	`FUT+IV3MS+IV+IVSUFF_MOOD:I`	
Gloss	will + he / it + allege / claim / testify + [ind.]	
Lemma	[`Aid~aEaY_1`]	اِدَّعَى

With PADT, the word's description hopes to be more intuitive and explicit, and yet to better explain the non-trivial underlying morphological process:

Form	`yadda'I`	*yaddaʿī* يَدَّعِي	`sa`	*sa* سَ
Morph	`"ya" >>| FtaCI |<< "u"`		`"sa"`	
Tag	`VIIA-3MS--`		`F---------`	
Form	`idda'Y`	*iddaʿā* اِدَّعَى	`sa`	*sa* سَ
Morph	`IFtaCY`		`"sa"`	
Root	`"d ' w"`		`"s"`	
Reflex	allege, claim, testify		*future marker*	
Class	verb		particle	

Irrespective of the weak character of the morphophonemic pattern, the suffixation of `|<< "u"` is common to all third person singular indicative imperfective verbs, plus others. Similarly for the subjunctive and jussive templates:

`"ya" >>| FtaCI |<< "a"`	`yadda'iya`	*yaddaʿiya* يَدَّعِيَ
`"ya" >>| FtaCI |<< ""`	`yadda'i`	*yaddaʿi* يَدَّعِ

Compare that with the prototypical regular conjugation:

`"ya" >>| FCuL |<< "u"`	`yaktubu`	*yaktubu* يَكْتُبُ
`"ya" >>| FCuL |<< "a"`	`yaktuba`	*yaktuba* يَكْتُبَ
`"ya" >>| FCuL |<< ""`	`yaktub`	*yaktub* يَكْتُبْ

Unlike the Buckwalter analyzer, ElixirFM is suited for both morphological analysis and generation, and can be used as an advanced multi-purpose morphological model. In the interactive mode, one can invoke various utility functions for lookup in the lexicon, inflection and derivation of lexemes, resolution of strings, exporting and pretty-printing of the information, etc., as well as explore the definitions of the underlying linguistic rules and data being involved. The ElixirFM source code and the lexicon itself are highly reusable by both computers and humans, cf. Figure 1.

Morphological disambiguation of Arabic encompasses subproblems like tokenization, 'part-of-speech' and full morphological tagging, lemmatization, diacritization, restoration of the structural components of words, and combinations thereof. Given a list of possible readings of an input string produced by an analyzer, it can be worthwhile to organize the analyses into a MorphoTrees hierarchy (Smrž and Pajas, 2004) with the string as its root and the full tokens as the leaves, grouped by their lemmas, non-vocalized canonical forms, and partitionings of the string into such forms. The shift from lists to trees enables clearer presentation of the options and their more convenient annotation. MorphoTrees promote gradual focusing on the solution through efficient top-down search or bottom-up pruning using restrictions on the properties of the nodes, and allow inheritance, refinement and sharing of information.

MorphoTrees in Figure 2 depict a complex annotation of the string `fhm` فهم resolved as *fa-hum* 'so they'. Alternative ways of annotating and details on the automation of some of the steps in the process are explained in (Smrž, 2007b).

3. Dependency Syntax

Morphological annotations identify the textual forms of a discourse lexically and recognize their grammatical properties. The analytical syntactic processing describes the superficial dependency structures in the discourse, whereas the tectogrammatical representation reveals the underlying dependency structures and restores the linguistic meaning.

Annotations on the analytical level are represented by dependency trees. Their nodes map, one to one, to the tokens resulting from the morphological analysis and tokenization, and their roots group the nodes according to the division into sentences or paragraphs. Edges in the trees show that there is a syntactic relation between the governor and its dependent, or rather, the whole subtree under and including the dependent. The nature of the government is expressed by the analytical functions of the nodes being linked.

Figures 3 and 4 analyze the following sentences:

> ... بعد أن أمضى فيها نحو عشرين عاما.
>
> '... after he had spent in it almost twenty years.'
>
> في ملف الأدب طرحت المجلة قضية اللغة العربية والأخطار التي تهددها.
>
> 'In the section on literature, the magazine presented the issue of the Arabic language and the dangers that threaten it.'

The connection between the PADT dependency analytical trees and the phrase-structure trees of PATB was studied in

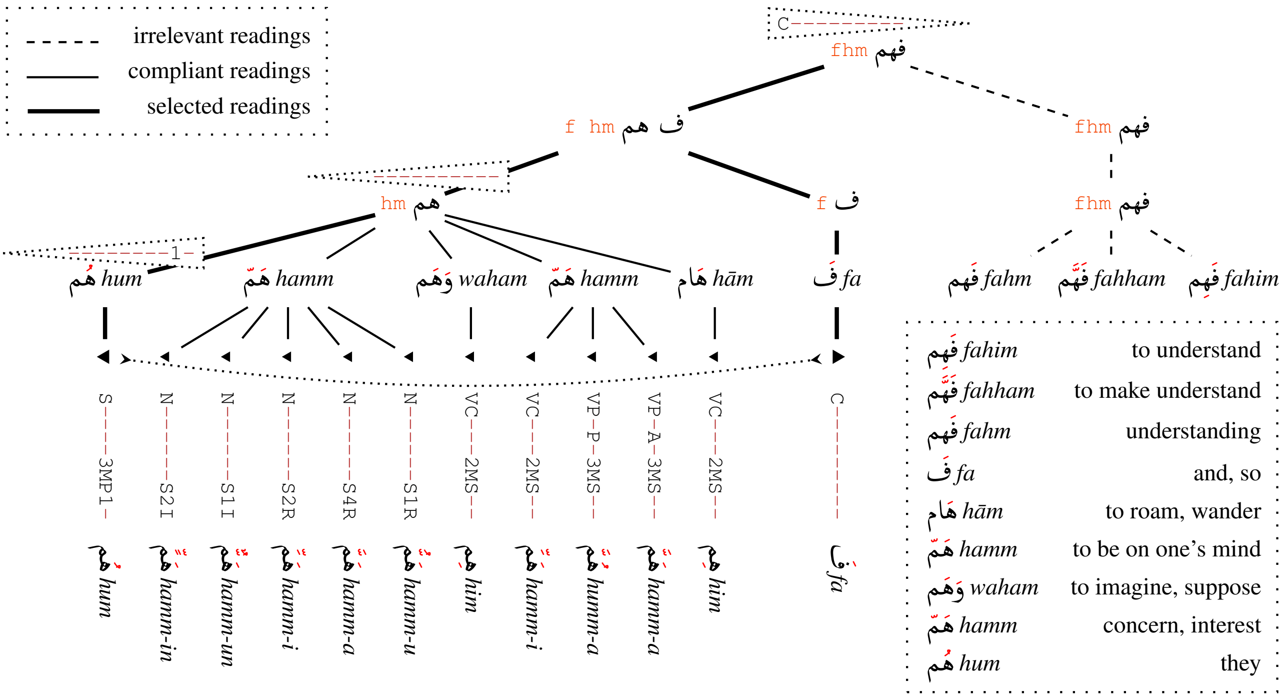


Figure 2: MorphoTrees of the orthographic string fhm فهم including annotation with restrictions. The dashed lines indicate there is no solution suiting the inherited restrictions in the given subtree. The dotted arc symbolizes that there can be implicit morphosyntactic constraints between the adjacent tokens in the analyses, the consistency of which should be verified.

(Žabokrtský and Smrž, 2003). Other notable insights are given in (Habash and Rambow, 2004; Habash et al., 2007).

The conversion from phrase-structure trees to dependencies needs to translate the topology of the original data and assign new labels to the nodes of the resulting trees. In some implementations, these tasks are strictly decoupled, and the translation rests in appointing the head subtree out of the children of a particular type of phrase. As illustrated below, this pure percolation mechanism attaching non-head subtrees to the head cannot account for all desired reconfigurations, requiring yet another kind of a translation procedure to be developed and applied on the temporary result.

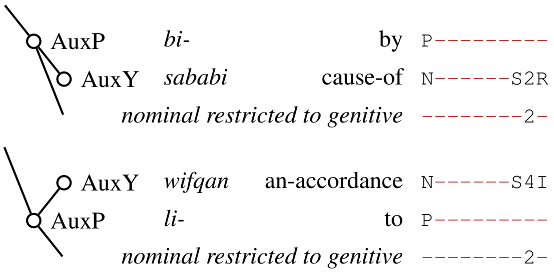
Our implementation of the conversion consists of more consecutive phases as well, but it attempts to use a stronger and more uniform formalism in both the ConDep primary phase and the DepDep secondary phase. Individual conversion rules specify subsequences of nodes and a subroutine that should be triggered if their pattern appears in the data. A rule's subroutine can manipulate the matching nodes rather freely and can return more than one transformed subtree to be integrated into the larger result, allowing more diverse attachments than what pure percolation achieves.

In Figure 3, we present a rule that transforms a SBAR containing a preposition *baʿda*, a conjunction *ʾan*, and some unspecified other children, in our case the adverbial S clause. This rule would also match on any uninflected preposition followed by *ʾanna*. The subroutine calls the recursive ConDep procedure on all the children, attaches the conjunction to the preposition and the rest of the nodes to the conjunction, and assigns the analytical labels AuxP and AuxC to the respective nodes. The preposition with its subtree is then returned. Other SBAR rules would replace the . . . .

One of the differences between PATB and PADT that must be taken into account when converting the syntactic data is the formal treatment of Arabic compound prepositions, such as *bi-sababi* 'because of', *wifqan li-* 'according to', *bi-'n-nisbati ʾilā* 'with respect to'. While PATB does not explicitly distinguish this phenomenon, the PADT approach inspired by the dependency formalism of the Prague Dependency Treebank for Czech does provide a formal solution to these and similar cases (Hajič et al., 1999: 162–163).

The criteria for regarding multi-word expressions as compound prepositions are that they be well established (usually listed in dictionaries) and not make sense when standing alone without a following nominal phrase. Compound prepositions were gradually gathered during annotations of the PADT analytical syntax, and can as well be extracted from the data. Their lists have been used in the conversion procedure of PATB and in annotation consistency checks.

The reattachments of the nodes in the compounds are implemented in the DepDep dependency tree 'parser', which matches on nodes in their linear order, but retains, accesses, and possibly modifies their dependency information.


AuxP bi- by P---------
AuxY sababi cause-of N------S2R
nominal restricted to genitive --------2-
AuxY wifqan an-accordance N------S4I
AuxP li- to P---------
nominal restricted to genitive --------2-

If a certain compound preposition consists of a preposition and a noun regardless of their word order, e.g. *bi-sababi* or *wifqan li-*, the noun as well as the following complement always depend on the preposition, which becomes the head

```
'SBAR'  =>  [ [

  [ ["P-------4-", "ba`da|qabla"],
    ["C---------", "'an"],
    undef ],

  [ ["P---------", ".+"],
    ["C---------", "'anna"],
    undef ],

  sub { my ($root, undef, @data) = @_;

    my @node = map { ConDep($_) } @data;

    PasteNode($node[1], $node[0]);

    PasteNode($_, $node[1]) foreach
                  @node[2 .. @node - 1];

    $node[0]->{'afun'} = "AuxP";
    $node[1]->{'afun'} = "AuxC";

    return $node[0] }

  ], . . . ]
```

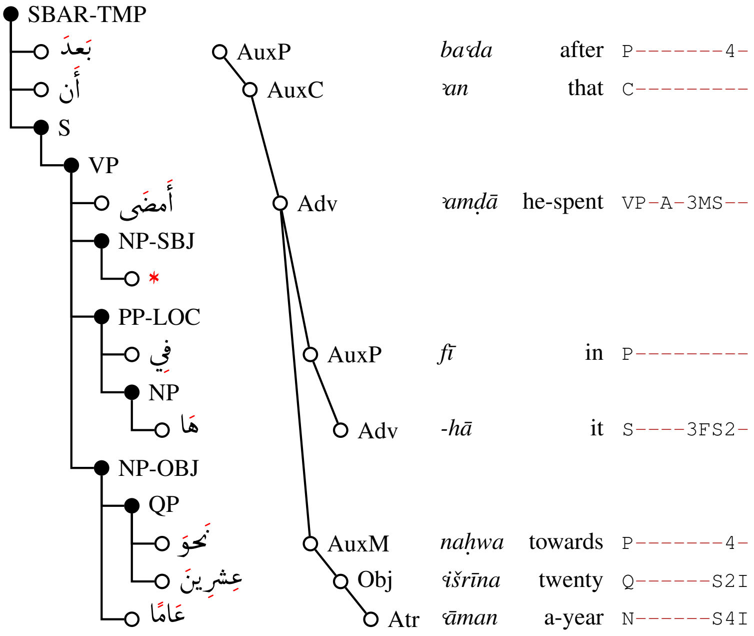
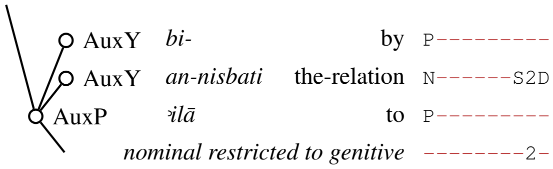
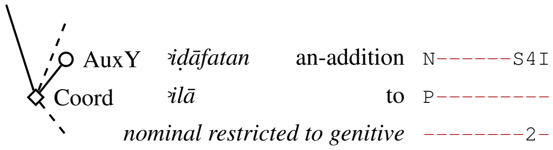


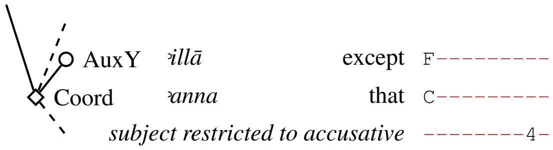
Figure 3: ConDep conversion rule applied to the top level of the phrase-structure tree, and the resulting dependency tree.

AuxP of that prepositional phrase. The nominal part of the compound preposition bears the auxiliary function AuxY, whereas the complement receives its analytical function according to the role of the whole prepositional phrase in a sentence. If a compound preposition is composed of a noun and two prepositions, e.g. *bi-’n-nisbati ʾilā*, the last preposition in the string becomes the head AuxP, and the remaining components depend on it side by side bearing the auxiliary functions AuxY. The complement that follows this compound preposition also hangs on the head AuxP and bears the analytical function determined by the context.

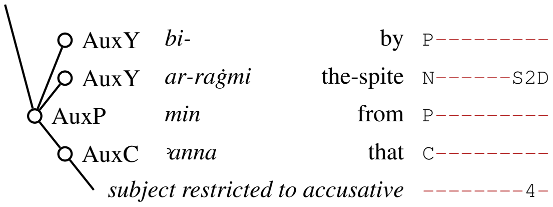


The motivation for this explicit identification of compound prepositions is that on the tectogrammatical layer, these compounds as well as all other synsemantic words disappear. Only the nodes of autosemantic words remain, representing their underlying syntactic and semantic roles.





Very similar to compound prepositions is our formal treatment of compound conjunctions, e.g. *ʾillā ʾanna* ‘however’, *ʾiḍāfatan ʾilā* ‘in addition to’. In these cases, one of the components is appointed the head (Coord in coordination, AuxC otherwise), while the other one attaches to it as AuxY. Multi-word expressions like *baʿda ʾan* ‘after’, *qabla ʾan* ‘before’, *bi-’r-raġmi min ʾanna* ‘in spite of the fact that, although’ are not considered to be compound conjunctions. They are regarded as prepositions or compound prepositions followed by a conjunction, due to the fact that the conjunction and its clause can in general be replaced by a nominal phrase.

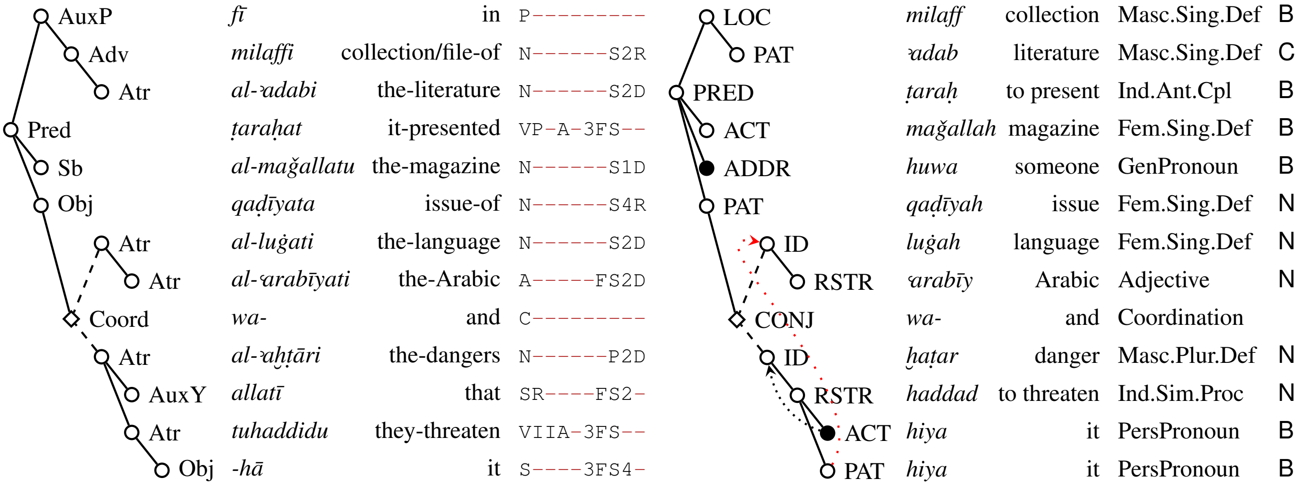


There are many other kinds of syntactic differences between PATB and PADT. Some structures in PATB tend to be more semantically oriented, while others are rather simplified. Note e.g. the use of QP in Figure 3, which breaks the grammatical dependency between the modifier *naḥwa*, the numeral, and the counted object (cf. Habash et al., 2007). The favored PATB annotation would, however, correspond to the tectogrammatical treatment of quantifiers in PADT.

Both ConDep and DepDep tackle even issues due to certain inconsistency of annotations. Flat phrases must be parsed, improper dependencies eliminated, incorrect instances of subordination need to be restructured to coordination, etc.

The functional labels in PATB capture several types of adverbials, conflated to Adv on the analytical level. On the other hand, the set of tectogrammatical functors in PADT is yet much more refined (cf. Mikulová et al., 2006).

The complete tree conversion process includes also the resolution of traces. This phase yields pointers of grammatical coreference, present in the manual analytical data as well (Hajič et al., 2004b). The recovery of textual coreference is performed on the tectogrammatical level, cf. Figure 4.



AuxP	*fī*	in	P---------
Adv	*milaffi*	collection/file-of	N------S2R
Atr	*al-ʾadabi*	the-literature	N------S2D
Pred	*ṭaraḥat*	it-presented	VP-A-3FS--
Sb	*al-mağallatu*	the-magazine	N------S1D
Obj	*qaḍīyata*	issue-of	N------S4R
Atr	*al-luġati*	the-language	N------S2D
Atr	*al-ʿarabīyati*	the-Arabic	A-----FS2D
Coord	*wa-*	and	C---------
Atr	*al-ʾaḫṭāri*	the-dangers	N------P2D
AuxY	*allatī*	that	SR----FS2-
Atr	*tuhaddidu*	they-threaten	VIIA-3FS--
Obj	*-hā*	it	S----3FS4-

LOC	*milaff*	collection	Masc.Sing.Def	B
PAT	*ʾadab*	literature	Masc.Sing.Def	C
PRED	*ṭaraḥ*	to present	Ind.Ant.Cpl	B
ACT	*mağallah*	magazine	Fem.Sing.Def	B
ADDR	*huwa*	someone	GenPronoun	B
PAT	*qaḍīyah*	issue	Fem.Sing.Def	N
ID	*luġah*	language	Fem.Sing.Def	N
RSTR	*ʿarabīy*	Arabic	Adjective	N
CONJ	*wa-*	and	Coordination	
ID	*ḫaṭar*	danger	Masc.Plur.Def	N
RSTR	*haddad*	to threaten	Ind.Sim.Proc	N
ACT	*hiya*	it	PersPronoun	B
PAT	*hiya*	it	PersPronoun	B

Figure 4: *Left:* Example of analytical annotation. Orthographic words are tokenized into lexical words, and their inflectional morphosyntactic properties are encoded using the positional tags. Coordination members are depicted with dashed edges. *Right:* Example of tectogrammatical annotation with resolved coreference (extra arcs) and indicated values of contextual boundness. Lexemes are identified by lemmas, and selected grammatemes are shown in place of morphosyntactic features.

4. Tectogrammatics

Tectogrammatics, the underlying syntax reflecting the linguistic meaning of an utterance, is the highest level of annotation in the family of Prague Dependency Treebanks (Hajič et al., 2006). It captures dependency and valency (Žabokrtský, 2005) with respect to the deep linguistic relations of discourse participants. In its generality, the description also includes topic–focus articulation, coreference resolution, and other non-dependency relations. The set of tectogrammatical annotations in PADT is still rather experimental, yet, we intend to develop this formalism further.

The topology of a tectogrammatical representation of a sentence is similar to that of the analytical level. In contrast to it, nodes in the tree may be deleted, inserted, and even reorganized. We speak of a transfer of structures from analytical to tectogrammatical, which can be partly automated. The nodes appear as lexical entries rather than inflected forms. Grammatemes, the deep grammatical parameters, abstract away from the morphological and analytical features of an utterance. Functors, the deep roles that the participants assume, include Actor, Patient, Addressee, Origin, Effect, various types of local and temporal modifications, Extent, Manner, Cause, Identity, Restriction, coordination types, and many more (Mikulová et al., 2006).

Figure 4 compares the analytical and tectogrammatical representations of a sentence. The inserted nodes are recovered from the discourse as the obligatory actants of the valency frames of the two verbs (cf. Bielický and Smrž, 2008). Values of contextual boundness, a feature from which the topic–focus dichotomy is inferred, are also indicated (cf. Hajičová and Sgall, 2003; Smrž and Hajič, 2008).

There is a number of structures on the analytical level that appear as a co-sign, i.e. their actual meaning results only from being combined. Quite often, one of the nodes in a structure is occupied by a verb while its other members are modifiers, which are not further developed.

The tectogrammatical level (t-level) erases some language-specific features present on the analytical level. The treatment of Arabic verbal negation is an instance thereof. Generally, verbal negation on the t-level is expressed by an abstract node for negation. The reason for it being a node and not a grammateme is that the deep ordering of the node with respect to the verb determines the scope of negation, with consequences for the information structure of a discourse.

In Arabic, a variety of combinations, such as *lam yaktub* vs. *mā kataba* 'he did not write', turn into exactly the same structures on the t-level. Stylistic variation is not reflected in tectogrammatics—it is believed that *lam yaktub* is more formal, while *mā kataba* is considered rather dialectal or used only in spoken discourse. The *mā* type of negation is used in rather fixed collocations, such as *mā zāla* 'still'.

The opposition of perfective *katab* and imperfective *yaktub* forms in Arabic, generally perceived as an opposition of past and non-past, is actually irrelevant for the deep notion of tense in this context. On the analytical level, the tense reference is expressed as a co-sign consisting of a negative particle and a finite verbal form. On the t-level, the tense indication is marked only in the grammatemes of the verbal node, i.e. [tense=Ant] for the anterior tense.

The other markers of verbal negation, *lā* and *lan*, are represented in the same manner, but relate to different tenses.

Structurally very similar on the analytical level is the use of a future marker for expressing positive posterior tense. In *sa-yaktubu* 'he will write', the modifier *sa-* is attached to the node of the verb, and itself has no offsprings. The more explicit form *sawfa* can also be used. In combination of *sawfa* with negation, the particle *lā* is inserted between the marker and the verb, i.e. *sawfa lā yaktubu*. However, *sa-* cannot combine with negation, in which case *lan* is used instead, as in *lan yaktuba* 'he will not write'. On the t-level, the modifier node containing the future marker is always deleted and the tense is indicated at the verb [tense=Post].

An interesting point is the treatment of *qad*, a modifying particle attached to verbs. When connected with the perfective form of a verb, it has the meaning of an aspectual nuance of completed action, like 'already'. On the t-level, this particle is deleted and the verbal node receives the grammatemes for anteriority and completeness [tense=Ant, aspect=Cpl]. However, when used with an imperfective verbal form, its meaning changes to possibility in the future, ‘it might well be that’. The grammatemes of the verb become [tense=Post, deontmod=Poss], but the modifier node is retained on the t-level, as is the case with other kinds of modality nodes (cf. Mikulová et al., 2006).

5. Quality Control

Our software environment for maintaining the PADT and PATB data is TrEd, an editor for tree-like structures developed in Perl by Petr Pajas. It is a highly customizable and programmable tool providing both the graphical user interface and the application programming interface used for network-oriented data processing, such as conversions or consistency checks (cf. Štěpánek, 2006).

TrEd integrates all the levels of annotation by enabling the user to invoke macros or external programs of any kind. During annotation, one can take great advantage of specific contexts/modes with predefined macro operations, keyboard-shortcuts, and stylesheets for informative display of the data. The dependency approach to syntax does not restrict TrEd itself. Next to MorphoTrees (Smrž and Pajas, 2004), we have for instance implemented contexts for viewing and possibly annotating phrase-structure trees.

Most of our quality control programs are written as TrEd scripts. Linguistic and formal constraints on the annotated data, as well as requirements on the mutual compatibility between levels, are implemented with transparent code reusable also in automatic tagging and partial parsing.

Using this technology, we have successfully discovered and eliminated many annotation errors and inconsistencies in both PADT and PATB, on all the levels of annotation. We will report on this process in detail in the documentation to PADT 2.0. The order of revisions is in tens of thousands of words, i.e. percents of the whole treebank, which means great improvement in the accuracy of PADT 2.0, and might have significant effect on the performance of any derived applications (cf. Habash et al., 2007).

Our experiments on an earlier version of the data indicated that annotating MorphoTrees is up to three times faster than disambiguating morphology in form of the classical MorphoLists. The inter-annotator disagreement on MorphoTrees was 5.3 %, on MorphoLists it reached 9.3 %. The inter-annotator disagreement in the attachment of nodes on the analytical level before revisions was measured to 9.2 %. With the upgrade of the morphological data in the whole treebank to ElixirFM, as well as with the introduction of our new tools for consistency verification and data processing, we believe that the inter-annotator agreement will yet considerably improve on these values.

6. Conclusion

In this contribution, we have overviewed the theoretical concepts behind the Prague Arabic Dependency Treebank, and have discussed converting the Penn Arabic Treebank into the PADT style. We have described the original data and the tools that we develop. PADT 2.0 will be an important new linguistic resource. TrEd with its extensions, ElixirFM, and Encode Arabic are open-source software published under the GNU General Public License:

```
http://ufal.mff.cuni.cz/~pajas/tred/
http://sf.net/projects/elixir-fm/
http://sf.net/projects/encode-arabic/
```

Acknowledgements

We would like to thank all our colleagues in the PADT project for their annotation efforts, inspiring ideas and valuable suggestions, and for their long-time support: Tereza Čečáková, Milada Frantová, Petr Pajas, Pavel Pecina, Zdeněk Žabokrtský, and others.

This research has been funded by the Ministry of Education of the Czech Republic (MSM0021620838 and MSM0021-620823) and by the Grant Agency of the Czech Academy of Sciences (1ET101120413).

References

Viktor Bielický and Otakar Smrž. Building the Valency Lexicon of Arabic Verbs. In *LREC 2008: Sixth Language Resources and Evaluation Conference*, Marrakech, Morocco, 2008.

Ann Bies. English-Arabic Treebank v 1.0. LDC2006T10, ISBN 1-58563-387-9, 2006.

Tim Buckwalter. Issues in Arabic Orthography and Morphology Analysis. In *COLING 2004 Computational Approaches to Arabic Script-based Languages*, pages 31–34, Geneva, 2004a.

Tim Buckwalter. Buckwalter Arabic Morphological Analyzer Version 1.0. LDC2002L49, ISBN 1-58563-257-0, 2002.

Tim Buckwalter. Buckwalter Arabic Morphological Analyzer Version 2.0. LDC2004L02, ISBN 1-58563-324-0, 2004b.

Simon Corston-Oliver, Anthony Aue, Kevin Duh, and Eric Ringger. Multilingual Dependency Parsing using Bayes Point Machines. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL)*, pages 160–167, Morristown, NJ, 2006.

Wolfdietrich Fischer. *A Grammar of Classical Arabic*. Yale Language Series. Yale University Press, third revised edition, 2002. Translated by Jonathan Rodgers.

Markus Forsberg and Aarne Ranta. Functional Morphology. In *Proceedings of the Ninth ACM SIGPLAN International Conference on Functional Programming, ICFP 2004*, pages 213–223. ACM Press, 2004.

David Graff. Arabic Gigaword Third Edition. LDC2007T40, 1-58563-460-3, 2007.

Nizar Habash and Owen Rambow. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, pages 573–580, Ann Arbor, Michigan, 2005.

Nizar Habash and Owen Rambow. Extracting a Tree Adjoining Grammar from the Penn Arabic Treebank. In *JEP-TALN 2004, Session Traitement Automatique de l’Arabe*, Fes, Morocco, April 2004 2004.

Nizar Habash and Owen Rambow. Arabic Diacritization through Full Morphological Tagging. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 53–56, Rochester, New York, April 2007. Association for Computational Linguistics.

Nizar Habash, Ryan Gabbard, Owen Rambow, Seth Kulick, and Mitch Marcus. Determining Case in Arabic: Learning Complex Linguistic Behavior Requires Complex Linguistic Features. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1084–1092, 2007.

Jan Hajič, Otakar Smrž, Petr Zemánek, Petr Pajas, Jan Šnaidauf, Emanuel Beška, Jakub Kráčmar, and Kamila Hassanová. Prague Arabic Dependency Treebank 1.0. LDC2004T23, ISBN 1-58563-319-4, 2004a.

Jan Hajič, Otakar Smrž, Petr Zemánek, Jan Šnaidauf, and Emanuel Beška. Prague Arabic Dependency Treebank: Development in Data and Tools. In *NEMLAR International Conference on Arabic Language Resources and Tools*, pages 110–117. ELDA, 2004b.

Jan Hajič, Otakar Smrž, Tim Buckwalter, and Hubert Jin. Feature-Based Tagger of Approximations of Functional Arabic Morphology. In *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories (TLT 2005)*, pages 53–64, Barcelona, Spain, 2005.

Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, and Marie Mikulová. Prague Dependency Treebank 2.0. LDC2006T01, ISBN 1-58563-370-4, 2006.

Jan Hajič et al. A Manual for Analytic Layer Annotation of the Prague Dependency Treebank. Technical Report 28, UFAL MFF UK, Charles University in Prague, 1999.

Eva Hajičová and Petr Sgall. Dependency Syntax in Functional Generative Description. In *Dependenz und Valenz – Dependency and Valency*, volume I, pages 570–592. Walter de Gruyter, 2003.

Seth Kulick, Ryan Gabbard, and Mitch Marcus. Parsing the Arabic Treebank: Analysis and Improvements. In *Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories (TLT 2006)*, pages 31–42, Prague, Czech Republic, 2006.

Xiaoyi Ma. Arabic English Parallel News Part 1. LDC2004T18, ISBN 1-58563-310-0, 2004.

Mohamed Maamouri and Ann Bies. Developing an Arabic Treebank: Methods, Guidelines, Procedures, and Tools. In *COLING 2004 Computational Approaches to Arabic Script-based Languages*, pages 2–9, Geneva, 2004.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Hubert Jin. Arabic Treebank: Part 2 v 2.0. LDC2004T02, ISBN 1-58563-282-1, 2004a.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *NEMLAR International Conference on Arabic Language Resources and Tools*, pages 102–109. ELDA, 2004b.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Hubert Jin. Arabic Treebank: Part 1 v 3.0. LDC2005T02, ISBN 1-58563-330-5, 2005a.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, Hubert Jin, and Wigdan Mekki. Arabic Treebank: Part 3 v 2.0. LDC2005T20, ISBN 1-58563-341-0, 2005b.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, Hubert Jin, and Wigdan Mekki. Arabic Treebank: Part 4 v 1.0. LDC2005T30, ISBN 1-58563-343-7, 2005c.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330, 1993.

Marie Mikulová et al. A Manual for Tectogrammatical Layer Annotation of the Prague Dependency Treebank. Technical Report 30, UFAL MFF UK, Charles University in Prague, 2006.

Rani Nelken and Stuart M. Shieber. Arabic Diacritization Using Finite-State Transducers. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 79–86, Ann Arbor, 2005.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, 2007.

Karin C. Ryding. *A Reference Grammar of Modern Standard Arabic*. Cambridge University Press, 2005.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. D. Reidel & Academia, 1986.

Noah A. Smith, David A. Smith, and Roy W. Tromble. Context-Based Morphological Disambiguation with Random Fields. In *Proceedings of HLT/EMNLP 2005*, pages 475–482, Vancouver, 2005. Association for Computational Linguistics.

Otakar Smrž. ElixirFM — Implementation of Functional Arabic Morphology. In *ACL 2007 Proceedings of the Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pages 1–8, Prague, Czech Republic, June 2007a. Association for Computational Linguistics.

Otakar Smrž. *Functional Arabic Morphology. Formal System and Implementation*. PhD thesis, Charles University in Prague, 2007b.

Otakar Smrž and Jan Hajič. The Other Arabic Treebank: Prague Dependencies and Functions. In Ali Farghaly, editor, *Arabic Computational Linguistics: Current Implementations*. CSLI Publications, 2008.

Otakar Smrž and Petr Pajas. MorphoTrees of Arabic and Their Annotation in the TrEd Environment. In *NEMLAR International Conference on Arabic Language Resources and Tools*, pages 38–41. ELDA, 2004.

Jan Štěpánek. Post-annotation Checking of Prague Dependency Treebank 2.0 Data. In *Proceedings of the 9th International Conference TSD 2006*, number 4188 in Lecture Notes in Computer Science, pages 277–284. Springer-Verlag, 2006.

Gregory T. Stump. *Inflectional Morphology. A Theory of Paradigm Structure*. Cambridge Studies in Linguistics. Cambridge University Press, 2001.

Jim Yaghi and Sane Yagi. Systematic Verb Stem Generation for Arabic. In *COLING 2004 Computational Approaches to Arabic Script-based Languages*, pages 23–30, Geneva, 2004.

Zdeněk Žabokrtský. *Valency Lexicon of Czech Verbs*. PhD thesis, Charles University in Prague, 2005.

Zdeněk Žabokrtský and Otakar Smrž. Arabic Syntactic Trees: from Constituency to Dependency. In *EACL 2003 Conference Companion*, pages 183–186, Budapest, Hungary, 2003.

Imed Zitouni, Jeffrey S. Sorensen, and Ruhi Sarikaya. Maximum Entropy Based Restoration of Arabic Diacritics. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 577–584, Sydney, Australia, July 2006. Association for Computational Linguistics.