# Automatic Annotation of Morpho-Syntactic Dependencies in a Modern Hebrew Treebank[*]

Noemie Guthmann Yuval Krymolowski Adi Milea Yoad Winter

*Technion – Israel Institute of Technology* *Technion and Utrecht University*

## Introduction

Morpho-syntactic dependencies between sentence constituents are an inseparable part of syntactic analysis, in particular in Semitic languages. In those languages, because of the relatively free order of certain constituents, morpho-syntactic agreement features are sometimes the main clue for computational parsing models (Tsarfaty and Sima'an 2007,2008). However, despite their centrality for syntactic analysis, morpho-syntactic dependencies have so far not been annotated in Hebrew resources. In particular, such dependencies were not annotated in early versions of the Modern Hebrew Treebank (*MHT*, Sima'an et al. 2001), which to date is the only publicly available resource with syntactic analyses for Modern Hebrew. By developing a method for automatically adding dependency annotations to a Modern Hebrew treebank, the MHT project has aimed to contribute to treebank development for Semitic languages as well as for other languages.

This paper describes the development and implementation of the morpho-syntactic dependency scheme used in the MHT project. We concentrate on *mother-daughter dependencies*, in which the morphological features of one or more daughter nodes affect the morphological features or syntactic analysis of the mother node. The annotation scheme for such dependencies is based on familiar but non-trivial assumptions about grammatical rules for Modern Hebrew. These rules are used for two purposes. The first purpose is to annotate the mother-daughter dependencies between nodes in the treebank. The second purpose is to use the generated dependencies for annotating morpho-syntactic features of compound constituents. The rules are described in XML format and implemented using Python scripts. The scripts were run on a recent version of the MHT to produce dependency annotations in the MHT2, the most recent version of the treebank. A sample of the annotated dependencies in MHT2 was manually evaluated, which showed high accuracy of the automatic scheme. Errors detected mostly resulted from errors in the original syntactic annotation of the MHT, without the dependency annotations. Thus, the development of the dependency scheme and its automatic implementation also proved helpful in improving the quality of the manual annotation.

Similarly to the earlier versions of MHT, also one of the major syntactic resources for Arabic, the Penn Arabic Treebank (ATB, Maamouri

[*]We are grateful to Roy Bar-Haim, Alon Itai, Khalil Sima'an and Reut Tsarfaty for helpful discussions. Dalia Bojan's technical help was invaluable. This work was done at Technion as part of *Mila, Knowledge Center for Processing Hebrew*, http://mila.cs.technion.ac.il/english/index.html

et. al. 2004) does not include manual annotations of morpho-syntactic dependencies. Mother-daughter dependencies are also missing from another important Arabic resource - the Prague Arabic Dependency Treebank (PADT, Smrž et al. 2008). The annotation schemes used for the MHT and the ATB are designed in close correspondence to the phrase structure grammar underlying the English Penn Treebank (Marcus et al., 1994). Because of this similarity between the MHT and the ATB, and since the rules for morpho-syntactic dependencies are very similar in Modern Hebrew and Standard Arabic, we expect our methodology for automatic annotation to be generally applicable to other "Penn-compatible" syntactic resources like the ATB. This expectation is further supported by cross-linguistic relations between NLP works on Hebrew and Arabic dealing with POS-tagging (Bar-Haim et al., 2007, Mansour et al. 2007). Resources for other Semitic languages like Amharic (Alemu et al. 2003) and Ugaritic (Zemánek 2007) may also benefit from our annotation methodology.

After giving some background on the MHT in section 1, this paper describes the significance of morpho-syntactic dependencies in Modern Hebrew (section 2), and the dependency annotation scheme developed for the MHT (section 3). The rule mechanism that was used for automatically annotating dependencies in the MHT is described in section 4, and its manual evaluation is described in section 5.

## 1. Background – the Modern Hebrew Treebank

Semitic words are formed by concatenating a word stem and several clitics, including some prepositions, conjunctions, and the definiteness marker. These word parts are referred to as *word segments*. Word segments are assigned part-of-speech (POS) tags similar to POS tags of Indo-European words, with a fixed order of the possible POSs within the Semitic word. Word limits do not necessarily correspond to constituency structure: the segments of a single word may belong to several different constituents within the sentence. Thus, morphological analysis, and word segmentation in particular, is a precondition for syntactic analysis in Semitic languages (Sima'an et al. 2001, Diab et al. 2004). The Modern Hebrew Treebank (MHT) includes segmented and syntactically annotated text from *Ha'aretz* daily newspaper. The texts in the corpus cover several domains (news, society, politics, sports and business). Version 2 of the treebank (MHT2) is comprised of 6,501 sentences, 123,446 word tokens, and 162,829 word segments. Sections of text were automatically segmented into sentences and fed into an automatic morphological disambiguator for Hebrew (Segal, 2000), which outputs a suggested morphological analysis for each Hebrew word. This morphological annotation was automatically converted to word segments, with a POS tag assigned to each word-segment (Bar-Haim et al., 2007). This pre-processed text was then syntactically analyzed by human annotators, whose task was twofold: (i) correcting the automatically derived segmentation and POS-tagging; (ii) producing a syntactic analysis using the *Semtags* annotation-aiding tool (Bonnema, 1997). Following this stage, the last manually annotated version of the MHT was automatically enhanced by marking mother-daughter dependencies, which is in the focus of the present paper. The resulting treebank is publicly available[1] as *MHT2*.

A pilot syntactic annotation of 500 Hebrew sentences is described in Sima'an et al. (2001). Much of the annotation scheme in this pilot version of the MHT was maintained in MHT2, with some modifications. The MHT scheme applies flat annotations for structures where the Hebrew syntax allows a relaxed order of constituents. Notably, both VP complements and adjuncts are analyzed as VP external. Besides these departures from English grammatical conventions, many of the POS tags and annotation conventions were used in a similar way to the conventions of the Penn Treebank (Marcus et al., 1994).

## 2. The significance of mother-daughter dependencies in Hebrew syntactic analysis

The main new aspect of the annotation scheme of MHT2 is the analysis of "*mother-daughter*" dependencies: these dependencies are manifested through the (recursive) percolation of morphological information from one or more daughter nodes to their mother. We distinguish two different processes:

- *Dependency annotation* – marking a node according to the role it plays in determining its mother's features.
- *Feature percolation* – using these dependencies for feature percolation from daughter to mother.

In English, the head constituent often determines the phrase's morphological features that must agree with the features of other constituents. Parsing systems often exploit such dependencies for disambiguation in head-driven methods (Charniak, 2001; Collins, 2003). In Semitic languages, due to some "free order" aspects of their syntax, mother-daughter dependencies are often syntactically critical. The Hebrew verb and its subject, internal arguments and adjuncts do not appear in a strict order in the sentence. As a result, agreement features percolated via mother-daughter dependencies often help to identify and disambiguate grammatical relations. For a recent demonstration that morpho-syntactic dependencies and agreement features significantly improve the performance of a non-lexicalized parser for Hebrew, see Tsarfaty and Sima'an (2007, 2008).

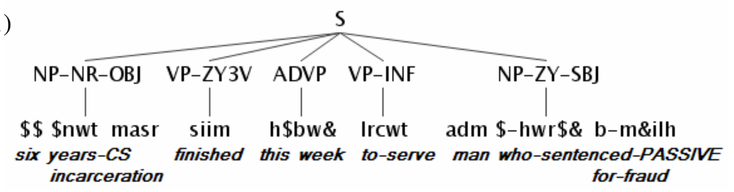
Mother-daughter dependencies in Semitic languages are often complicated by the *construct state* (CS) case configuration within complex nominals. The special properties of Hebrew CS nominals are well-documented in the linguistic literature (Glinert 1989, Wintner 2000, Danon 2008). These constructions exhibit a syntactic relation between two adjacent nominals. The first part of the CS is the *construct noun*, which carries a special CS morphology. According to standard tests of selectional restrictions, the construct noun functions as the "semantic head" of the CS nominal. The second, obligatory, part in a CS configuration is the *post-*

[1] http://www.mila.cs.technion.ac.il/english/resources/corpora/treebank/ver2.0/index.html

*construct NP*, which follows the CS noun. The head noun of the post-construct NP is morphologically unmarked. The number and gender features of CS nominals are determined by the construct noun. Importantly, however, syntactic definiteness is determined by the post-construct NP. Such dependencies on more than one daughter are common in MHT2, and in particular with construct states: 23.3% of all mother-daughter feature percolations in MHT2 are multiple daughter percolations, 30.8% of those multiple dependencies are in CS nominals.

As a simple example, consider the following Hebrew sentence with the morpho-syntactic annotation of its main constituents:[2]

(1)



"A man sentenced for fraud finished serving this week six years of incarceration"

Sentence (1) contains two candidate NPs for being the subject of the matrix verb *siim* ("finished"). The main factor for classifying the second NP ("a man sentenced for fraud") as the subject of (1) is its singular masculine feature ('ZY'), which agrees with the singular masculine morphologiy of the verb *siim* ("finished"). The 'ZY' feature of the subject, in turn, percolates from its head noun *adm* ("man"/"person"). In our work, this morpho-syntactic dependency relation between the subject NP node and the head noun is automatically recognized. Subsequently, the agreement features of the subject are automatically percolated from the manually annotated noun in the MHT.

## 3. The dependency annotation scheme

We mark mother-daughter dependencies using *dependency tags* that are added as additional features of syntactic categories. These tags encode the features that are percolated from a constituent to its mother node. Our scheme includes six types of dependency tags, listed and described in table 1.

| Dependency tag | ***Used for marking:*** |
|---|---|
| DEP_HEAD | Percolation of all the agreement features |
| DEP_MAJOR | Percolation of all features except for definiteness or number features marked on sister nodes |
| DEP_DEFINITE | Percolation of the definiteness feature |
| DEP_NUMBER | Percolation of the number feature |
| DEP_ACCUSATIVE | Percolation of accusative case from the accusative marker 'AT' |
| DEP_HEAD_MULTIPLE | Percolation of agreement features from multiple sisters (for conjunctions) |

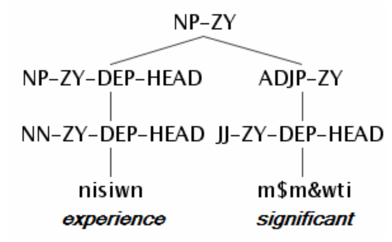
**Table 1: Morpho-Syntactic Dependency Tags in MHT2**

[2] For our notation of agreement features and the Hebrew transliteration see tables A and B in the appendix.

**DEP_HEAD:**

The tag DEP_HEAD is used for annotating the daughter node from which all features percolate to the mother node. In example (2) below, the words "experience" (*nisiwn*) and "significant" (*m$m&wti*) are heads of their respective NP and ADJP: they pass on their features, Z (masculine) and Y (singular), to the dominating phrasal node. In turn, the lower-level NP is tagged DEP_HEAD, since all of its morphological features are passed on to the main NP.

(2)

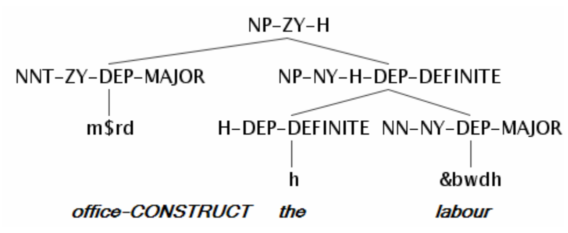


"A significant experience"

**DEP_MAJOR and DEP_DEFINITE:**

The tag DEP_MAJOR is used for annotating the daughter node from which most but not all features percolate to the mother node. In such structures, some of the features are percolated from nodes other than the head. This is the case in the aforementioned construct state nominals. We use the DEP_MAJOR tag to denote the percolation of all features but definiteness from the construct noun. The definiteness feature is inherited from the post-construct NP, which is annotated with the DEP_DEFINITE tag.

(3)



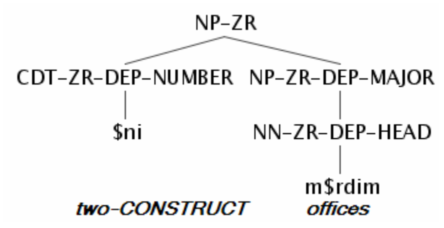
"The Labour Office"

Example (3) illustrates another use of the DEP_MAJOR dependency tag in our scheme, which is a result of the segmentation conventions of the MHT. As explained in Sima'an et al. (2001), the syntactic annotation scheme of the treebank analyzes definite articles as separate segments rather than as one of the nominal features of the head noun. Consequently, the definiteness feature of the definite noun *h-&bwdh* in (3) is inherited from the definiteness (*h*) segment. The other agreement features of this constituent are inherited from the noun *&bwdh*. The definiteness marker *h* is therefore annotated with the DEP_DEFINITE tag, and the noun *&bwdh* with the DEP_MAJOR tag.

**DEP_NUMBER:**

The dependency tag DEP_NUMBER is used for annotating a daughter node from which only the number feature percolates to the mother node. Consider the following example, which illustrates the annotation of feature dependency within a numeral CS.

(4)



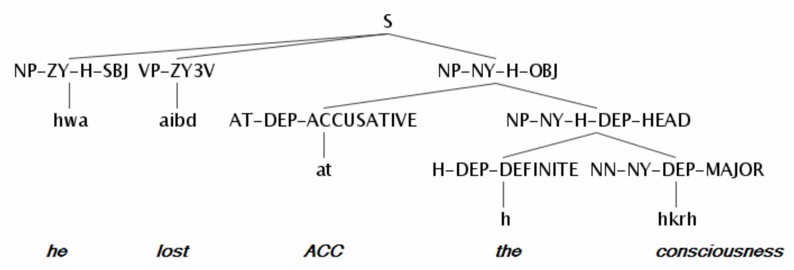
"Two offices"

The numeral item *$ni* ("two") has the marked construct morphology. However, unlike the nominal construct state in (3), this construct provides only the number feature (indicated by the DEP_NUMBER dependency tag); the head element *m$rdim* ("offices") determines the other features of the noun phrase (gender and definiteness), and is therefore tagged DEP_MAJOR.

**DEP_ACCUSATIVE:**

The tag DEP_ACCUSATIVE denotes the accusative marker 'AT', which requires the mother NP to be an object (have the OBJ feature). The following example illustrates this dependency tag.

(5)

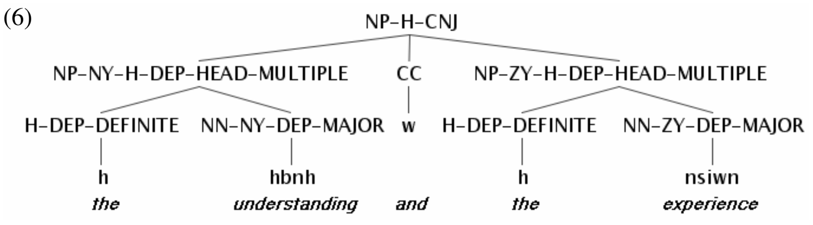


"He lost consciousness"

**DEP_ HEAD_MULTIPLE:**

The dependency tag DEP_HEAD_MULTIPLE marks feature percolation from multiple coordinated constituents to the mother node. The features percolated depend upon the nature of the phrasal constituent. In an adjective phrase conjunction, the features of the conjoined adjectives have to match each other, and therefore all features are percolated from daughter adjectives to the mother conjunction with no conflict. In a nominal conjunction, however, the gender and number may differ between the coordinated noun phrases. Therefore only the definiteness feature is percolated in nominal conjunctions, as in the following example:

(6)



"The understanding and the experience"

This example illustrates a limitation of the current annotation scheme: in such conjunctions, the scheme could have kept track of the fact that the dominating conjunction node inherits its plural and masculine feature values from both conjuncts. Such complex dependencies are not analyzed in MHT2.

## 4. Implementing the scheme: automatic dependency annotation and feature percolation

The annotation scheme presented in the previous section was manually coded as rules in XML format. These rules describe marking of mother-daughter dependencies in noun phrases, verb phrases, adjective phrases and prepositional phrases in the MHT. Further, the XML rules were processed by Python scripts that used them for adding morpho-syntactic dependencies and feature percolation to the MHT. The output of these scripts is the current MHT2. See Appendix C for some figures on the MHT2 treebank.

*The annotation procedure*

A linguist annotator divided the phrasal categories of the corpus into a small number of general cases. Each case defines the appropriate dependency scheme from those in Table 1, and the feature percolation that it sanctions. Overall, 24 dependency rules were written in XML format. Most of the rules were designed for NPs, where feature percolation depends on heterogeneous factors: especially the presence of a definite article, pronouns or proper names, coordination, embedding and construct states. After a pilot version of these rules was run on the corpus, the output was manually checked. Most of the rules were corrected accordingly, and subsequently run again on a second iteration to produce the current version of MHT2.

In the original annotation scheme, some special structural cases were defined as exceptions to the dependency rules. 1,942 such cases were automatically extracted from the corpus and examined by a linguist annotator. However, in effect it turned out that only a few of these cases were incorrectly annotated by the automatic procedure. These errors were manually corrected. The estimated time for these manual corrections was 30 hours.

*XML rules*

The rules were coded in XML format and processed by Python scripts. The scripts applied the rules in a bottom-up order, so that to allow the percolation of features from daughter to mother nodes within structures. In total, 24 rules were developed. Consider for example the following informal rule describing dependency annotation and feature percolation for construct state NPs:

Upon matching an NP which is comprised of a construct state noun (category 'NNT') followed by an embedded NP (the "post-construct" NP):

- Copy all the properties of the construct noun except definiteness to the mother NP node
- Set the dependency tag of the construct noun to 'DEP_MAJOR'
- Copy the definiteness of the post-construct NP node to the mother NP node
- Set the dependency of the post-construct NP node to 'DEP_DEFINITE'

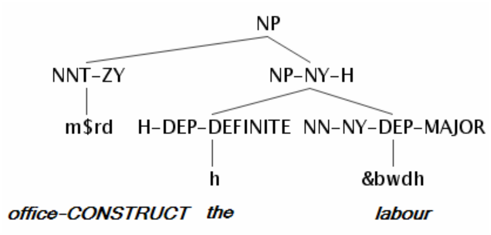
The XML code corresponding to this rule is given below:

```
<rule name='5.1' dest='father'>
 <struct>
  <father label='NP'>
    <child1 label='NNT' position='1' dep='DEP_MAJOR'>
      <copy>-def</copy>
    </child1>
    <child2 label='NP' position='1' dep='DEP_DEFINITE'>
      <copy>def</copy>
    </child2>
  </father>
 </struct>
</rule>
```

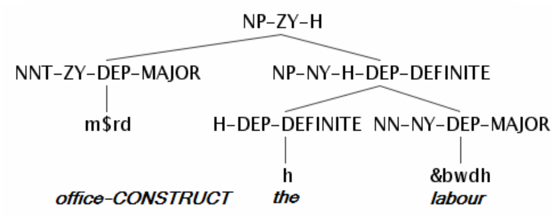
By applying the Python scripts to this XML description of the rule and to structure (7) below, we generate analysis (8) for example (3) above.

In (7) the construct noun *m$rd* ("office") is masculine singular ('ZY'). The post-construct NP, *h&bwdh* ("the labour") has already been automatically processed by another rule in our scheme that assigns dependency features to its constituents and percolates the feminine gender ('N'), singular number ('Y') and definiteness ('H') features up to the phrase node. The construct state rule above generates (8) by percolating the masculine singular features ('ZY') of the construct noun to the NP, while copying definiteness ('H') from the post-construct NP.

(7)



(8)



"The Labour Office"

## 5. Correctness evaluation of dependency annotations

Evaluation of our dependency annotation was performed by selecting at random 50 sentences from the MHT2 (about 0.77% of the corpus' sentences). A linguist who was not involved in the rule-writing process checked these sentences for morpho-syntactic annotation, dependency annotation, feature percolation and syntactic agreement between constituents.

The 50 sentences contained on average 19.2 words and 25.3 word segments per sentence. Of the 1,227 dependency tags annotation in the analyses of these sentences, 837 dependency tags were on word segments and 390 on complex phrases. The errors that were detected were either in manual annotations or in the automatic application of the dependency scheme.

1. Incorrect *manual* POS or syntactic tag annotations – 3 errors.
2. Incorrect *automatic* dependency tag annotation – 6 errors.
3. Underspecification of features – 6 cases.

In addition, one incorrect percolation of features was detected in one of the "exceptional" cases mentioned above, where percolation was manually revised. Also the 6 incorrect dependency tag annotations were caused by mistakes in the manual syntactic annotation of a previous version of the MHT. Cases of feature underspecification were mainly due to the partial coverage of the rules on nominal conjunctions. Another cause of feature underspecification was the percolation in certain structures of the bi-gender feature ("B") to the upper mother node.

From this error analysis we conclude that the automatic part of the annotation is highly reliable, and errors are only likely to happen when mistakes in manual annotation are detected.

## 6. Conclusion

While manual annotation of treebanks is known to be time consuming and error prone, a syntactically annotated corpus can be augmented with additional data based on linguistically informed schemes. In this work we examined one such case – dependency annotations and feature percolation in a Modern Hebrew treebank. The complexities of Semitic morpho-syntax, and especially the construct state, make this task considerably challenging. However, our conclusion from this work is that once syntactic constituency and categories are reliably annotated, augmenting them with dependency tags can be done using a robust automatic procedure, informed by solid linguistic analysis of the relevant constructions. We therefore believe that our contribution is not only useful for syntactically informed tasks in Modern Hebrew, but that it could also be of high value to on-going work on other Semitic resources, and notably syntactic resources for Arabic like the Penn Arabic Treebank.

## Appendices

### A. Agreement features

| | |
|---|---|
| Gender | ***Z=masculine, N=feminine, B=both*** |
| Number | Y=singular, R=plural, B=both |
| Person | 1,2,3 |
| Tense | V=past, H=present, T=future, C=imperative |
| Definiteness | H=definite, U=underspecified |

### B. Hebrew transliteration table

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E & | F $ | G | H | I | J @ | K | L | M | N | O | P | Q | R |
| א | ב | צ | ד | ע | ש | ג | ה | י | ט | כ | ל | מ | נ | % | פ | ק | ר |
| S | T | U | W | X | Z | | | | | | | | | | | | |
| ס | ת | " | ו | ח | ז | | | | | | | | | | | | |

### C. MHT2 figures

| | Words | Segments | Segment dependencies | ***Structural dependencies*** |
|---|---|---|---|---|
| For total number of sentences | 123,446 | 162,829 | 107,251 | 48,955 |
| Average per sentence | 19.0 | 25.0 | 16.5 | 7.5 |

**Table 2: figures about the MHT2 (6,501 sentences)**

| Percolation levels | Number of features percolated | ***Percentage*** |
|---|---|---|
| 1 | 124,844 | 72.035% |
| 2 | 38,872 | 22.429% |
| 3 | 8,116 | 4.683% |
| 4 | 1,268 | 0.731% |
| 5 | 198 | 0.114% |
| 6 | 12 | 0.007% |
| **Total** | 173,310 | |

**Table 3: Distribution of the number of percolation levels**

| Dependency tag | Total number | ***Average per sentence*** |
|---|---|---|
| DEP_HEAD | 103,062 | 16.0 |
| DEP_MAJOR | 25,600 | 4.0 |

**Table 4: Number of MAJOR vs HEAD dependencies**

| Percolation from | | Number of rule applications | Percentage (out of total number of rule applications) | ***Percentage (out of total number of rule applications involving feature percolation)*** |
|---|---|---|---|---|
| **No percolation** | | 2,776 | 2% | - |
| **Single child** | | 93,192 | 69% | 76.7% |
| **Multiple children** | *Total:* | 28,413 | 21.1% | 23.3% |
| | *Construct states:* | 8,775 | 6.5% | 7,2% |
| | *Conjunctions:* | 1,761 | 1.3% | 1,5% |

**Table 5: Distribution of rule applications**

## References

A. Alemu, L. Asker and G. Eriksson . 2003. An Empirical Approach to Building an Amharic Treebank. *TLT 2003*. http://w3.msi.vxu.se/~rics/TLT2003/doc/alemu_et_al.pdf

R. Bar-Haim, K. Sima'an, and Y. Winter. 2007. Part-of-Speech Tagging of Modern Hebrew Text. In *Journal of Natural Language Engineering*.

R. Bonnema. 1997. Data Oriented Semantics. Master's thesis.

M. Collins. 2003. Head-Driven Statistical Models for Natural Language Parsing. *Computational Linguistics*, 29(4).

E. Charniak. 2001. Immediate-Head Parsing for Language Models. In *Proceedings of ACL 2001*: 116-123.

G. Danon. 2008. Definiteness spreading in the Hebrew construct state. *Lingua* 118(7), 872-906.

M Diab, K. Hacioglu, and D. Jurafsky. 2004. Automatic tagging of arabic text: From raw text to base phrase chunks. In Marcu, Dumais and Salim Roukos, editors, *HLT-NAACL 2004*: Short Papers, pages 149-152.

L. Glinert. 1989. The Grammar of Modern Hebrew. *Cambridge University Press*.

M. Johnson. 1998. PCFG Models of Linguistic Tree Representations. *Computational Linguistics*, 24(4):613–632.

M. Maamouri, A. Bies, T. Buckwalter, and W. Mekki. 2004. The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In *NEMLAR International Conference on Arabic Language Resources and Tools*.

S. Mansour, K. Sima'an and Y. Winter. 2007. Smoothing a Lexicon-based POS Tagger for Arabic and Hebrew. *ACL2007 Workshop on Computational Approaches to Semitic Languages.*

M. Marcus, G. Kim, M. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn Treebank: Annotating Predicate-Argument Structure.

E. Segal. 2000. Hebrew Morphological Analyzer for Hebrew undotted texts. Master's thesis.

K. Sima'an, A. Itai, Y. Winter, A. Altman, and N. Nativ. 2001. Building a Tree-Bank of Modern Hebrew Text. In *Traitment Automatique des Langues*.

O. Smrž, V. Bielický, I. Kouřilová, J. Kráčmar, J. Hajič, P. Zemánek. 2008. Prague Arabic Dependency Treebank: A Word on the Million Words. In *Proceedings of the LREC-2008 workshop on HLT & NLP within the Arabic world: Arabic Language and local languages processing*.

R. Tsarfaty, and K. Sima'an. 2007. Accurate Unlexicalized Parsing for Modern Hebrew. In *Proceedings of Text, Speech and Dialog* (TSD).

R. Tsarfaty and K. Sima'an. 2007. "Three-Dimensional Parametrization for Parsing Morphologically Rich Languages". In *Proceedings of the International Conference on Parsing Technologies* (IWPT).

R. Tsarfaty and K. Sima'an. 2008. Relational-Realizational Parsing. In *Proceedings of The 22nd International Conference on Computational Linguistics (CoLing).*

S. Wintner. 2000. Definiteness in the Hebrew Noun Phrase. In *Journal of Linguistics*, 36:319-363.

P. Zemánek. 2007. A Treebank of Ugaritic. Annotating Fragmentary Attested Languages. In *TLT 2007.* http://tlt07.uib.no/papers/9.pdf