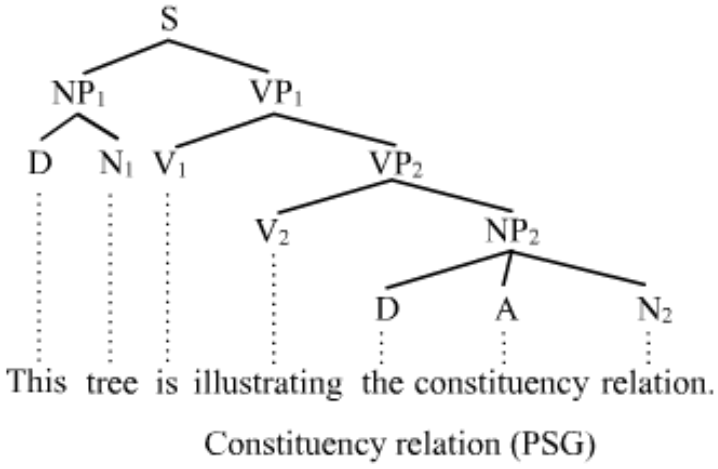
Parser und Tagger

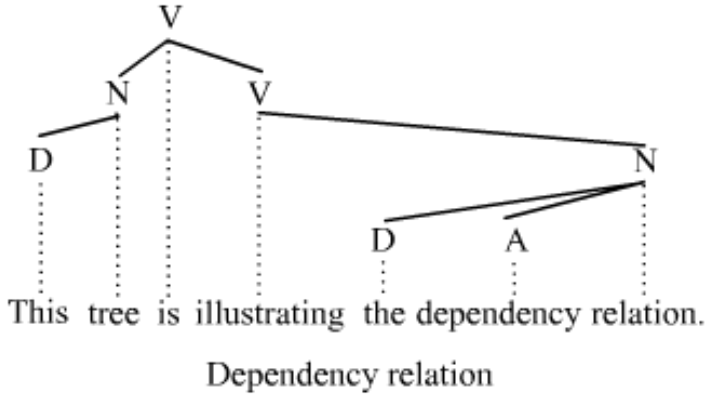
MaltParser

- Deterministischer Dependenz-Parser, implementiert in Java (seit Version 1.0)
- Parsing-Modell wird aus einer Treebank induziert
- Neue Daten werden mit dem induzierten Modell geparst
- Webseite: `http://maltparser.org/download.html`
- Auf unseren Servern unter `/resources/processors/parser/maltparser-1.8/`
- Modelle für Deutsch, Englisch, Französisch und Schwedisch
- Eingabe und Ausgabe im *CoNLL Format*

Dependenz- vs. Konstituenten-Parser



Constituency relation (PSG)

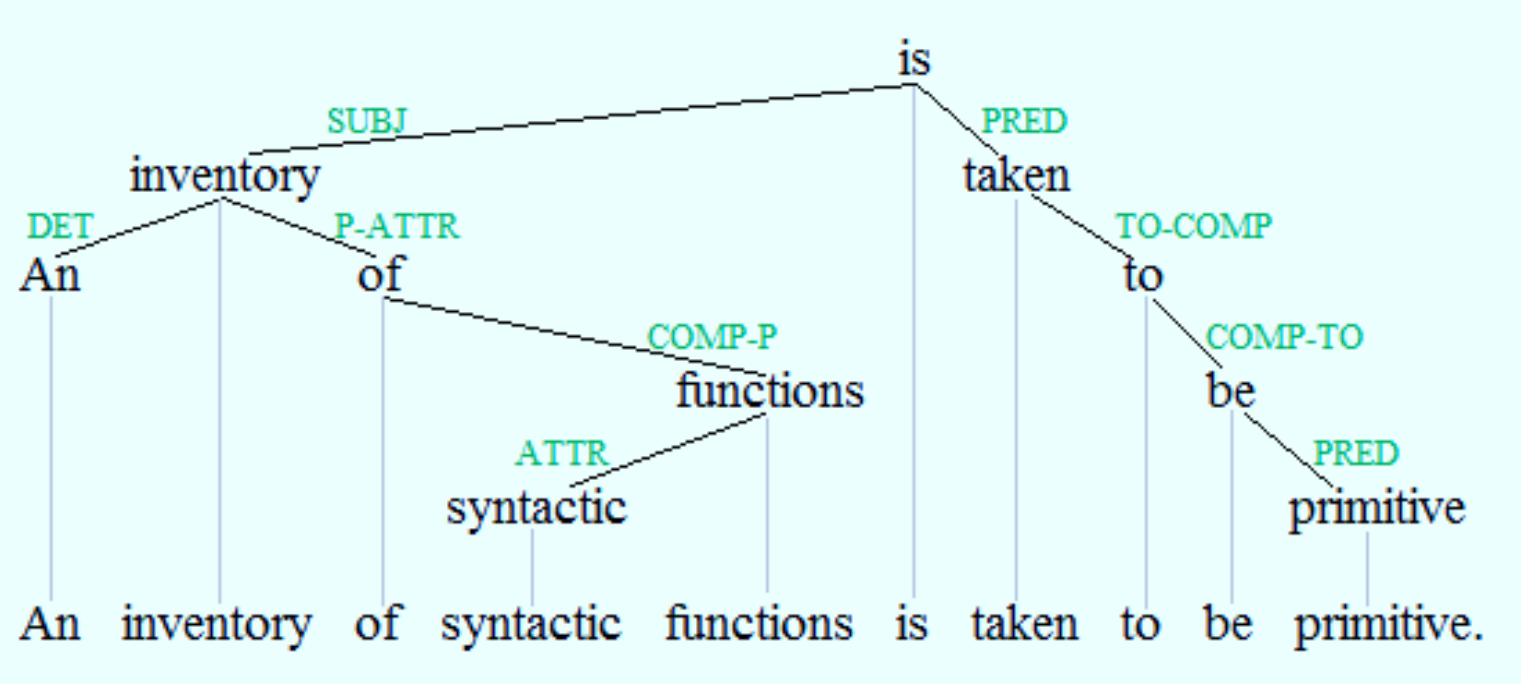


Dependency relation

Quelle:

`https://en.wikipedia.org/wiki/Phrase_structure_grammar`

Dependenzbäume mit Kantenbeschriftungen



Quelle:

`https://commons.wikimedia.org/w/index.php?curid=21976793`

CoNLL Format

- Standardisiertes Format für die Eingaben/Ausgaben bei CoNLL Shared Task 2007
- Text-Dateien mit der Erweiterung `.conll`
- Eine Zeile besteht aus 10 durch `TAB` getrennten Feldern
- Ein Satz besteht aus einem oder mehreren Tokens, je eines auf einer Zeile
- Eine Datei kann mehrere Sätze enthalten (getrennt durch eine Leerzeile)
- Leerzeichen in den Feldern sind nicht erlaubt!

`http://universaldependencies.org/format.html`

CoNLL Format - Felder

1. `ID` - Der Zähler für Tokens (beginnt mit 1 in jedem Satz)
2. `FORM` - Wortform
3. `LEMMA` - Lemma des Wortes
4. `CPOSTAG` - POS-Tag (sprachabhängig)
5. `POSTAG` - Detailliertes POS-Tag (sprachabhängig)
6. `FEATS` - Syntaktische und/oder morphologische Merkmale (sprachabhängig)
7. `HEAD` - Head-Token für das aktuelle Token (ID oder 0)
8. `DEPREL` - Dependenzrelation mit Head-Token (von den Treebank-Annotationen abhängig)

CoNLL Format - Bemerkungen

- Die folgenden Felder *müssen* immer ausgefüllt werden:
  - Bei den Trainingsdaten:
    `ID`, `FORM`, `CPOSTAG`, `POSTAG`, `HEAD`, `DEPREL`
  - Bei den Testdaten:
    `ID`, `FORM`, `CPOSTAG`, `POSTAG`
- Andere Felder können mit dem Platzhalter „_" besetzt werden
- Falls es keine detaillierten POS-Tags für eine bestimmte Sprache gibt, sind die Felder `CPOSTAG` und `POSTAG` gleich
- Mehrere Merkmale werden durch das Symbol „|" getrennt
- Dependenzrelation für den Startknoten ist `ROOT` (mit dem `HEAD`-Wert 0)

CoNLL Format - Beispiel

Satz aus einem Trainingsset

1	Trouble	_	NN	NN	_	2	SBJ	_	_
2	is	_	VBZ	VBZ	_	0	ROOT	_	_
3	,	_	,	,	_	2	P	_	_
4	she	_	PRP	PRP	_	5	SBJ	_	_
5	has	_	VBZ	VBZ	_	2	PRD	_	_
6	lost	_	VBN	VBN	_	5	VC	_	_
7	it	_	PRP	PRP	_	6	OBJ	_	_
8	just	_	RB	RB	_	6	MNR	_	_
9	as	_	RB	RB	_	8	AMOD	_	_
10	quickly	_	RB	RB	_	8	AMOD	_	_
11	.	_	.	.	_	2	P	_	_

CoNLL Format - Beispiel

Satz aus einem Testset

1	Pierre	_	NNP	NNP	_
2	Vinken	_	NNP	NNP	_
3	,	_	,	,	_
4	61	_	CD	CD	_
5	years	_	NNS	NNS	_
6	old	_	JJ	JJ	_
7	,	_	,	,	_
8	will	_	MD	MD	_
9	join	_	VB	VB	_
10	the	_	DT	DT	_
11	board	_	NN	NN	_
12	as	_	IN	IN	_
13	a	_	DT	DT	_
14	nonexecutive	_	JJ	JJ	_
15	director	_	NN	NN	_
16	Nov.	_	NNP	NNP	_
17	29	_	CD	CD	_
18	.	_	.	.	_

CoNLL Format - Beispiel

Andere Sprache

1	Grundavdraget	_	N	NN	DD|SS	2	SS	_	_
2	blir	_	V	BV	PS	0	ROOT	_	_
3	2500	_	R	RO	_	4	DT	_	_
4	kr	_	N	NN	_	2	SP	_	_
5	(	_	I	IR	_	0	ROOT	_	_
6	=	_	I	IG	_	15	DT	_	_
7	4500	_	R	RO	_	8	DT	_	_
8	minskat	_	P	TP	PA	0	ROOT	_	_
9	med	_	PR	PR	_	14	AA	_	_
10	1/5	_	N	NN	_	9	PA	_	_
11	av	_	PR	PR	_	10	ET	_	_
12	10000	_	R	RO	_	13	DT	_	_
13	kr	_	N	NN	_	11	PA	_	_
14	)	_	I	IR	_	0	ROOT	_	_
15	.	_	P	IP	_	2	IP	_	_

Erster Start des MaltParsers

- Starten des MaltParsers ohne Argumente:

```
:~$ java -jar /path/to/maltparser-1.8/maltparser-1.8.jar
```

  - Gibt die Hauptoptionen für den MaltParser aus

- Starten der Hilfe für den MaltParser:

```
:~$ java -jar /path/to/maltparser-1.8/maltparser-1.8.jar -h
```

  - Gibt alle Optionen für den MaltParser aus

Den MaltParser trainieren

```
:~$ java -jar /path/to/maltparser-1.8/maltparser-1.8.jar -c model \
-i training-file.conll -m learn
```

- **`model`** : Name des Modells
  Achtung: nur den Namen ohne Pfad angegeben, die Erweiterung (**`.mco`**) wird *automatisch* angefügt!
- **`training-file.conll`** : Datei mit den Trainingsdaten im CoNLL-Format
- **`-m learn`** : der Parser wird im Lern-Modus gestartet

Das erstellte Modell wird (hier) in einer Datei namens **`model.mco`** im aktuellen Verzeichnis gespeichert.

Vortrainierte Modelle für den MaltParser

- Für diejenigen, die nicht mit unterschiedlichen Algorithmen und Modellen experimentieren möchten, sondern nur einen Dependenzparser brauchen
- Aktuell sind drei fertige Modelle vorhanden: **`engmalt`**, **`fremalt`**, **`swemalt`** und **`espmalt`** (alle jeweils **`*.linear`** und **`*.poly`**)
- Download: `http://maltparser.org/mco/mco.html`
- Basieren auf Penn Treebank, French Treebank und Swedisch Treebank

Parsen von Daten im MaltParser

```
:~$ java -jar /path/to/maltparser-1.8/maltparser-1.8.jar -c model\
-i test-file.conll -o result-file.conll -m parse
```

- **`model`** : Name des Modells
  Achtung: nur den Namen ohne die Erweiterung (**`.mco`**) angeben
- **`test-file.conll`** : Datei mit den Testdaten
- **`result-file.conll`** : Datei mit den Ergebnissen des Parsers
- **`-m parse`** : der Parser wird im Parsing-Modus gestartet

MaltParser – Weitere Informationen

- Userguide: `http://maltparser.org/userguide.html`
- (Technische) Informationen zur Optimierung:
  `http://www.maltparser.org/guides/opt/quick-opt.pdf`

Übung 9