Syntactic Formalisms for Parsing Natural Languages

Aleš Horák, Miloš Jakubíček, Vojtěch Kovář
(based on slides by Juyeon Kang)

ia161@nlp.fi.muni.cz

Autumn 2013

Study materials

- Course materials and homeworks are available on the following web site:
  https://is.muni.cz/course/fi/autumn2011/IA161
- Refer to *Dependency Parsing*, Synthesis: Lectures on Human Language Technologies, S. kübler, R. McDonald and J. Nivre, 2009

Outline

- Introduction to Dependency parsing methods
- Dependency Parsers

Introduction to Dependency parsing

Motivation

a. dependency-based syntactic representation seem to be useful in many applications of language technology: machine translation, information extraction

$\rightarrow$ transparent encoding of predicate-argument structure

b. dependency grammar is better suited than phrase structure grammar for language with free or flexible word order

$\rightarrow$ analysis of diverse languages within a common framework

c. leading to the development of accurate syntactic parsers for a number of languages

$\rightarrow$ combination with machine learning from syntactically annotated corpora (e.g. treebank)

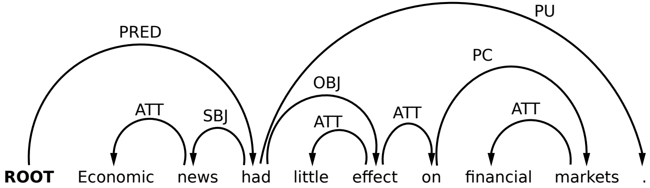
Introduction to Dependency parsing

Dependency parsing

"Task of automatically analyzing the dependency structure of a given input sentence"

Dependency parser

"Task of producing a labeled dependency structure of the kind depicted in the follow figure, where the words of the sentence are connected by typed dependency relations"



Definitions of dependency graphs and dependency parsing

Dependency graphs: syntactic structures over sentences

Def. 1.: A sentence is a sequence of tokens denoted by

$$S = w_0 w_1 \ldots w_n$$

Def. 2.: Let $R = \{r_1, \ldots, r_m\}$ be a finite set of *possible dependency relation types* that can hold between any two words in a sentence. A relation type $r \in R$ is additionally called an *arc label*.

Definitions of dependency graphs and dependency parsing

Dependency graphs: syntactic structures over sentences

Def. 3.: A dependency graph $G = (V, A)$ is a labeled directed graph, consists of nodes, V, and arcs, A, such that for sentence $S = w_0 w_1 \ldots w_n$ and label set R the following holds:

1. $V \subseteq \{w_0 w_1 \ldots w_n\}$
2. $A \subseteq V \times R \times V$
3. if $(w_i, r, w_j) \in A$ then $(w_i, r', w_j) \notin A$ for all $r' \neq r$

Approach to dependency parsing

a. **data-driven**
 it makes essential use of machine learning from linguistic data in order to parse new sentences

b. **grammar-based**
 it relies on a formal grammar, defining a formal language, so that it makes sense to ask whether a given input is in the language defined by the grammar or not.

$\rightarrow$ **Data-driven have attracted the most attention in recent years.**

Data-driven approach

according to the *type of parsing model* adopted,
the algorithms used to learn the model from data
the algorithms used to parse new sentences with the model

a. **transition-based**
start by defining a transition system, or state machine, for mapping a sentence to its dependency graph.

b. **graph-based**
start by defining a space of candidate dependency graphs for a sentence.

Data-driven approach

a. **transition-based**

- **learning problem:** induce a model for predicting the next state transition, given the transition history
- **parsing problem:** construct the optimal transition sequence for the input sentence, given induced model

b. **graph-based**

- **learning problem:** induce a model for assigning scores to the candidate dependency graphs for a sentence
- **parsing problem:** find the highest-scoring dependency graph for the input sentence, given induced model

Transition-based Parsing

- Transition system consists of a set C of parser configurations and of a set D of transitions between configurations.
- **Main idea:** a sequence of valid transitions, starting in the *initial configuration* for a given sentence and ending in one of several *terminal configurations*, defines a valid dependency tree for the input sentence.

$$D_{1'm} = d_1(c_1), \ldots, d_m(c_m)$$

Transition-based Parsing

- **Definition**
  Score of $D_{1'm}$ factors by configuration-transition pairs (c_i, d_i):

$$s(D_{1'm}) = \sum_{i=1}^{m} s(c_i, d_i)$$

- **Learning**
  Scoring function $s(c_i, d_i)$ for $d_i(c_i) \in D_{1'm}$
- **Inference**
  Search for highest scoring sequence $D^*_{1'm}$ given $s(c_i, d_i)$

Transition-based Parsing

Inference for transition-based parsing

- **Common inference strategies:**
  - Deterministic [Yamada and Matsumoto 2003, Nivre et al. 2004]
  - Beam search [Johansson and Nugues 2006, Titov and Henderson 2007]
  - Complexity given by upper bound on transition sequence length
- **Transition system**
  - Projective O(n) [Yamada and Matsumoto 2003, Nivre 2003]
  - Limited non-projective O(n) [Attardi 2006, Nivre 2007]
  - Unrestricted non-projective O(n2) [Nivre 2008, Nivre 2009]

Transition-based Parsing

Learning for transition-based parsing

- **Typical scoring function:**
  - $s(c_i, d_i) = w \cdot f(c_i, d_i)$ where $f(c_i, d_i)$ is a feature vector over configuration c_i and transition d_i and w is a weight vector $[w_i = \text{ weight of feature} f_i(c_i, d_i)]$
- **Transition system**
  - Projective O(n) [Yamada and Matsumoto 2003, Nivre 2003]
  - Limited non-projective O(n) [Attardi 2006, Nivre 2007]
  - Unrestricted non-projective O(n2) [Nivre 2008, Nivre 2009]
- **Problem**
  - Learning is local but features are based on the global history

Graph-based Parsing

- For a input sentence S we define a graph $G_s = (V_s, A_s)$ where $V_s = \{w_0, w_1, \ldots, w_n\}$ and $A_s = \{(w_i, w_j, l) | w_i, w_j \in V \text{ and } l \in L\}$
- Score of a dependency tree T factors by subgraphs $G_s, \ldots, Gs$:

$$s(T) = \sum_{i-1}^{m} s(G_i)$$

- Learning: **Scoring function** $s(G_i)$ for a subgraph $G_i \in T$
- Inference: Search for maximum spanning tree scoring sequence T^* of G_s given $s(G_i)$

Graph-based Parsing

Learning graph-based models

- **Typical scoring function:**
  - $s(G_i) = w \cdot f(G_i)$ where $f(G_i)$ is a high-dimensional feature vector over subgraphs and w is a weight vector
    $[w_j =$ weight of feature $f_j(G_i)]$
- **Structured learning** [McDonald et al. 2005a, Smith and Johnson 2007]:
  - Learn weights that maximize the score of the correct dependency tree for every sentence in the training set
- **Problem**
  - Learning is global (trees) but features are local (subgraphs)

Grammar-based approach

a. **context-free dependency parsing**
exploits a mapping from dependency structures to CFG structure representations and reuses parsing algorithms originally developed for CFG → chart parsing algorithms

b. **constraint-based dependency parsing**
- parsing viewed as a constraint satisfaction problem
- grammar defined as a set of constraints on well-formed dependency graphs
- finding a dependency graph for a sentence that satisfies all the constraints of the grammar (having the best score)

Grammar-based approach

a. **context-free dependency parsing**

Advantage: Well-studied parsing algorithms such as CKY, Earley's algorithm can be used for dependency parsing as well.

→ need to convert dependency grammars into efficiently parsable context-free grammars; (e.g. bilexical CFG, Eisner and Smith, 2005)

b. **constraint-based dependency parsing**

defines the problem as constraint satisfaction

- Weighted constraint dependency grammar (WCDG, Foth and Menzel, 2005)
- Transformation-based CDG

Dependency parsers

- **Trainable parsers**
  - Probabilistic dependency parser (Eisner, 1996, 2000)
  - MSTParser (McDonald, 2006)-graph-based
  - MaltParser (Nivre, 2007, 2008)-transition-based
  - K-best Maximum Spanning Tree Dependency Parser (Hall, 2007)
  - Vine Parser
  - ISBN Dependency Parser
- **Parsers for specific languages** defines the problem as constraint satisfaction
  - Minipar (Lin, 1998)
  - WCDG Parser (Foth *et al.*, 2005)
  - Pro3Gres (Schneider, 2004)
  - Link Grammar Parser (Lafferty *et al.*, 1992)
  - CaboCha (Kudo and Matsumoto, 2002)

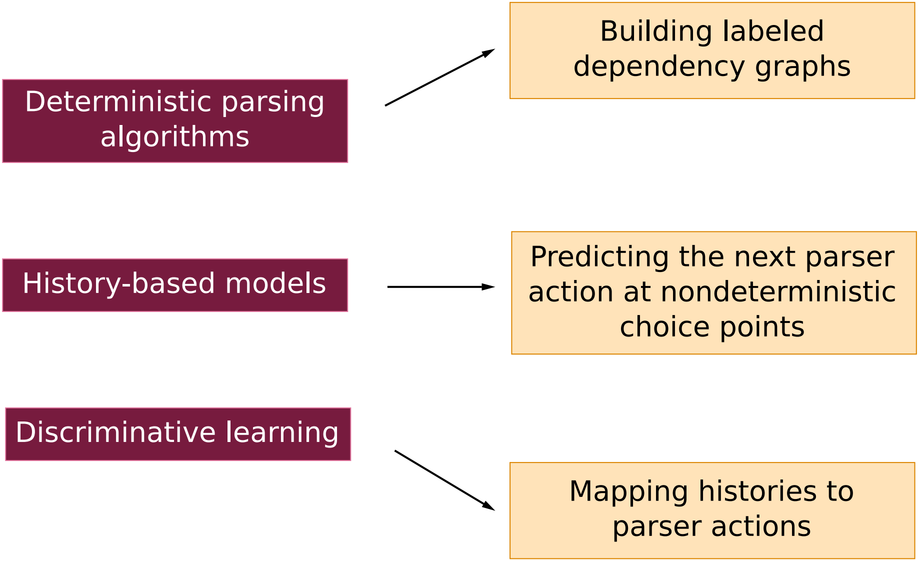
MaltParser

Data-driven dependency parsing system (Last version, 1.6.1, J. Hall, J. Nilsson and J. Nivre)

- Transition-based parsing system
- Implementation of inductive dependency parsing
- Useful for inducing a parsing model from treebank data
- Useful for parsing new data using an induced model

Useful links
http://maltparser.org

Components of system



MSTParser

Running system

- Input: part-of-speech tags or word forms

1	Den	_	PO	PO	DP	2	SS	_	_
2	blir	_	V	BV	PS	0	ROOT	_	_
3	gemensam	_	AJ	AJ	_	2	SP	_	_
4	für	_	PR	PR	_	2	OA	_	_
5	alla	_	PO	PO	TP	6	DT	_	_
6	inkomsttagare	_	N	NN	HS	4	PA	_	_
7	oavsett	_	PR	PR	_	2	AA	_	_
8	civilständ	_	N	NN	SS	7	PA	_	_
9	.	_	P	IP	_	2	IP	_	_

- Output: column containing a dependency label

MSTParser

Minimum Spanning Tree Parser (Last version, 0.2, R. McDonald et al., 2005, 2006)

- Graph-based parsing system

Useful links
http://www.seas.upenn.edu/ strctlrn/MSTParser/MSTParser.html

MSTParser

Running system

- Input data format:

w1	w2	. . .	wn
p1	p2	. . .	pn
l1	l2	. . .	ln
d1	d2	. . .	d2

Where,

- w1 ... wn are the n words of the sentence (tab deliminated)
- p1 ... pn are the POS tags for each word
- l1 ... ln are the labels of the incoming edge to each word
- d1 ... dn are integers representing the postition of each words parent

- Example:

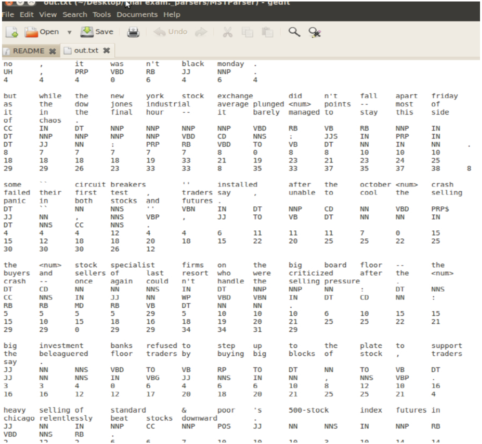
For example, the sentence ”John hit the ball” would be:

John	hit	the	ball
N	V	D	N
SBJ	ROOT		MOD OBJ
2	0	4	2

MSTParser

Running system

- Output: column containing a dependency label



Comparing parsing accuracy

Graph-based Vs. Transition-based MST Vs. Malt

Language	MST	Malt
Arabic	**66.91**	66.71
Bulgarian	**87.57**	87.41
Chinese	85.90	**86.92**
Czech	**80.18**	78.42
Danish	**84.79**	84.77
Dutch	**79.19**	78.59
German	**87.34**	85.82
Japanese	90.71	**91.65**
Portuguese	86.82	**87.60**
Slovene	**73.44**	70.30
Spanish	**82.25**	81.29
Swedish	82.55	**84.58**
Turkish	63.19	**65.68**
Average	**80.83**	80.75

Presented in *Current Trends in Data-Driven Dependency Parsing* by Joakim Nivre, 2009

Link Parser

Syntactic parser of English, based on the Link Grammar (version, 4.7.4, Feb. 2011, D. Temperley, D, Sleator, J. Lafferty, 2004)

- Words as blocks with connectors + or -
- Words rules for defining the connection between the connectors
- Deep syntactic parsing system

Useful links
http://www.link.cs.cmu.edu/link/index.html
http://www.abisource.com/

Link Parser

- Example of a parsing in the Link Grammar:

let's test our proper sentences!

http://www.link.cs.cmu.edu/link/submit-sentence-4.html

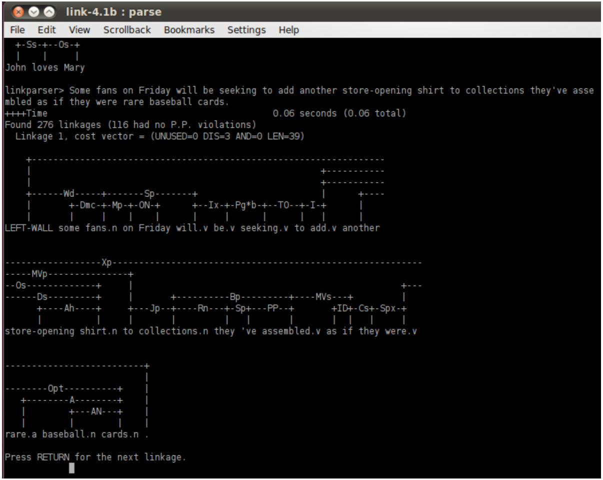
Link Parser

John gives a book to Mary.

```
Press RETURN for the next linkage.
linkparser> John gives a book to Mary.
++++Time                                          0.00 seconds (0.06 total)
Found 2 linkages (2 had no P.P. violations)
  Linkage 1, cost vector = (UNUSED=0 DIS=0 AND=0 LEN=9)

    +-----------------Xp----------------+
    |             +------MVp-----+      |
    |             +----Os---+    |      |
    +---Wd--+--Ss-+    +-Ds-+    +-Js+  |
    |       |     |    |    |    |   |  |
LEFT-WALL John gives.v a book.n to Mary .

Press RETURN for the next linkage.
linkparser>
  Linkage 2, cost vector = (UNUSED=0 DIS=1 AND=0 LEN=7)

    +-----------------Xp----------------+
    |             +----Os---+           |
    +---Wd--+--Ss-+    +-Ds-+-Mp-+-Js+  |
    |       |     |    |    |    |   |  |
LEFT-WALL John gives.v a book.n to Mary .

linkparser>
```

Link Parser

Some fans on Friday will be seeking to add another store-opening shirt to collections they’ve assembled as if they were rare baseball cards.



WCDG parser

Weighted Constraint Dependency Grammar Parser (version, 0.97-1, May, 2011, W. Menzel, N. Beuck, C. Baumgärtner)

- incremental parsing
- syntactic predictions for incomplete sentences
- Deep syntactic parsing system

Useful links
http://nats-www.informatik.uni-hamburg.de/view/CDG/ParserDemo