Introduction to Data-Driven Dependency Parsing

Introductory Course, ESSLLI 2007

Ryan McDonald\textsuperscript{1} Joakim Nivre\textsuperscript{2}

\textsuperscript{1}Google Inc., New York, USA
E-mail: ryanmcd@google.com

\textsuperscript{2}Uppsala University and Växjö University, Sweden
E-mail: nivre@msi.vxu.se
Introduction

- **Dependency parsing:**
  - Syntactic parsing using dependency-based representations.
- **Data-driven models:**
  - Models for dependency parsing based on machine learning.
Overview of the Course

- **Dependency parsing** (Joakim)
- Machine learning methods (Ryan)
- Transition-based models (Joakim)
- Graph-based models (Ryan)
- Loose ends (Joakim, Ryan):
  - Other approaches
  - Empirical results
  - Available software
Lecture 1: Outline

- Dependency syntax:
  - Basic concepts
  - Terminology and notation
  - Dependency graphs

- Dependency parsing:
  - Grammar-driven methods
  - Data-driven methods

- Pros and cons of dependency parsing
Dependency Syntax

- The basic idea:
  - Syntactic structure consists of lexical items, linked by binary asymmetric relations called dependencies.

- In the words of Lucien Tesnière [Tesnière 1959]:

  La phrase est un ensemble organisé dont les éléments constituant sont les mots. [1.2] Tout mot qui fait partie d’une phrase cesse par lui-même d’être isolé comme dans le dictionnaire. Entre lui et ses voisins, l’esprit aperçoit des connexions, dont l’ensemble forme la charpente de la phrase. [1.3] Les connexions structurales établissent entre les mots des rapports de dépendance. Chaque connexion unit en principe un terme supérieur à un terme inférieur. [2.1] Le terme supérieur reçoit le nom de régissant. Le terme inférieur reçoit le nom de subordonné. Ainsi dans la phrase Alfred parle […], parle est le régissant et Alfred le subordonné. [2.2]
The basic idea:

- Syntactic structure consists of lexical items, linked by binary asymmetric relations called dependencies.

In the words of Lucien Tesnière [Tesnière 1959]:

- The sentence is an organized whole, the constituent elements of which are words. [1.2] Every word that belongs to a sentence ceases by itself to be isolated as in the dictionary. Between the word and its neighbors, the mind perceives connections, the totality of which forms the structure of the sentence. [1.3] The structural connections establish dependency relations between the words. Each connection in principle unites a superior term and an inferior term. [2.1] The superior term receives the name governor. The inferior term receives the name subordinate. Thus, in the sentence Alfred parle [. . .], parle is the governor and Alfred the subordinate. [2.2]
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.

Diagram:
- Economic: nmod
- news: sbj
- had: obj
- little: effect
- on: financial
- markets:  
Economic news had little effect on financial markets.
### Terminology

<table>
<thead>
<tr>
<th>Superior</th>
<th>Inferior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Head</td>
<td>Dependent</td>
</tr>
<tr>
<td>Governor</td>
<td>Modifier</td>
</tr>
<tr>
<td>Regent</td>
<td>Subordinate</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>
## Terminology

<table>
<thead>
<tr>
<th>Superior</th>
<th>Inferior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Head</td>
<td>Dependent</td>
</tr>
<tr>
<td>Governor</td>
<td>Modifier</td>
</tr>
<tr>
<td>Regent</td>
<td>Subordinate</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>
Notational Variants

~had~

~news~

~Economic~

~effect~

~little~

~on~

~markets~

~financial~

Economic news had effect little on markets financial.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Economic news had little effect on financial markets.
Comparison

- Dependency structures explicitly represent
  - head-dependent relations (directed arcs),
  - functional categories (arc labels),
  - possibly some structural categories (parts-of-speech).

- Phrase structures explicitly represent
  - phrases (nonterminal nodes),
  - structural categories (nonterminal labels),
  - possibly some functional categories (grammatical functions).

- Hybrid representations may combine all elements.
Some Theoretical Frameworks

- Word Grammar (WG) [Hudson 1984, Hudson 1990]
- Functional Generative Description (FGD) [Sgall et al. 1986]
- Dependency Unification Grammar (DUG) [Hellwig 1986, Hellwig 2003]
- Meaning-Text Theory (MTT) [Mel’čuk 1988]
- Functional Dependency Grammar (FDG) [Tapanainen and Järvinen 1997, Järvinen and Tapanainen 1998]
- Topological/Extensible Dependency Grammar ([T/X]DG) [Duchier and Debusmann 2001, Debusmann et al. 2004]
Some Theoretical Issues

- Dependency structure sufficient as well as necessary?
- Mono-stratal or multi-stratal syntactic representations?
- What is the nature of lexical elements (nodes)?
  - Morphemes?
  - Word forms?
  - Multi-word units?
- What is the nature of dependency types (arc labels)?
  - Grammatical functions?
  - Semantic roles?
- What are the criteria for identifying heads and dependents?
- What are the formal properties of dependency structures?
Some Theoretical Issues

- Dependency structure *sufficient* as well as necessary?
- **Mono-stratal** or multi-stratal syntactic representations?
- What is the nature of lexical elements (nodes)?
  - Morphemes?
  - Word forms?
  - Multi-word units?
- What is the nature of dependency types (arc labels)?
  - Grammatical functions?
  - Semantic roles?
- What are the criteria for identifying heads and dependents?
- What are the formal properties of dependency structures?
Criteria for Heads and Dependents

Criteria for a syntactic relation between a head $H$ and a dependent $D$ in a construction $C$ [Zwicky 1985, Hudson 1990]:

1. $H$ determines the syntactic category of $C$; $H$ can replace $C$.
2. $H$ determines the semantic category of $C$; $D$ specifies $H$.
3. $H$ is obligatory; $D$ may be optional.
4. $H$ selects $D$ and determines whether $D$ is obligatory.
5. The form of $D$ depends on $H$ (agreement or government).
6. The linear position of $D$ is specified with reference to $H$.

Issues:

- Syntactic (and morphological) versus semantic criteria
- Exocentric versus endocentric constructions
## Some Clear Cases

<table>
<thead>
<tr>
<th>Construction</th>
<th>Head</th>
<th>Dependent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exocentric</td>
<td>Verb</td>
<td>Subject (saj)</td>
</tr>
<tr>
<td></td>
<td>Verb</td>
<td>Object (obj)</td>
</tr>
<tr>
<td>Endocentric</td>
<td>Verb</td>
<td>Adverbial (vmod)</td>
</tr>
<tr>
<td></td>
<td>Noun</td>
<td>Attribute (nmod)</td>
</tr>
</tbody>
</table>

Economic news suddenly affected financial markets.
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

I can see that they rely on this and that.
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

I can see that they rely on this and that.
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

I can see that they rely on this and that.
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

```
I can see that they rely on this and that.
```

Diagram:
```
sbj
\arrow{\downarrow} I
\hphantom{can}
\hphantom{see}
\hphantom{that}
\hphantom{they}
\hphantom{rely}
\hphantom{on}
\hphantom{this}
\hphantom{and}
\hphantom{that}
\hphantom{.}
```
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

Introduction to Data-Driven Dependency Parsing 15(1)
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

I can see that they rely on this and that.
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

I can see that they rely on this and that.
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation

Diagram:

```
  | sbar |
  |      |
  v  obj
  ^  ^    ^
  sbj  vg  subj
  l  can  see  that  they  rely  on  this  and  that

?```

I can see that they rely on this and that.
Some Tricky Cases

- Complex verb groups (auxiliary ↔ main verb)
- Subordinate clauses (complementizer ↔ verb)
- Coordination (coordinator ↔ conjuncts)
- Prepositional phrases (preposition ↔ nominal)
- Punctuation
A dependency structure can be defined as a directed graph $G$, consisting of
- a set $V$ of nodes (vertices),
- a set $A$ of arcs (directed edges),
- a linear precedence order $<$ on $V$ (word order).

Labeled graphs:
- Nodes in $V$ are labeled with word forms (and annotation).
- Arcs in $A$ are labeled with dependency types:
  - $L = \{l_1, \ldots, l_{|L|}\}$ is the set of permissible arc labels.
  - Every arc in $A$ is a triple $(i, j, k)$, representing a dependency from $w_i$ to $w_j$ with label $l_k$. 

Introduction to Data-Driven Dependency Parsing 16(1)
Dependency Graph Notation

- For a dependency graph \( G = (V, A) \)
- With label set \( L = \{l_1, \ldots, l_{|L|}\} \)
  - \( i \rightarrow j \equiv \exists k : (i, j, k) \in A \)
  - \( i \leftrightarrow j \equiv i \rightarrow j \lor j \rightarrow i \)
  - \( i \rightarrow^* j \equiv i = j \lor \exists i' : i \rightarrow i', i' \rightarrow^* j \)
  - \( i \leftrightarrow^* j \equiv i = j \lor \exists i' : i \leftrightarrow i', i' \leftrightarrow^* j \)
Formal Conditions on Dependency Graphs

- **G** is (weakly) connected:
  - If \( i, j \in V \), \( i \leftrightarrow^* j \).

- **G** is acyclic:
  - If \( i \rightarrow j \), then not \( j \rightarrow^* i \).

- **G** obeys the single-head constraint:
  - If \( i \rightarrow j \), then not \( i' \rightarrow j \), for any \( i' \neq i \).

- **G** is projective:
  - If \( i \rightarrow j \), then \( i \rightarrow^* i' \), for any \( i' \) such that \( i < i' < j \) or \( j < i' < i \).
Connectedness, Acyclicity and Single-Head

Intuitions:
- Syntactic structure is complete (Connectedness).
- Syntactic structure is hierarchical (Acyclicity).
- Every word has at most one syntactic head (Single-Head).

Connectedness can be enforced by adding a special root node.

Economic news had little effect on financial markets.
Connectedness, Acyclicity and Single-Head

- Intuitions:
  - Syntactic structure is complete (Connectedness).
  - Syntactic structure is hierarchical (Acyclicity).
  - Every word has at most one syntactic head (Single-Head).
- Connectedness can be enforced by adding a special root node.

```
p
   pred
    nmod  sbj
      nmod
root

   obj
     nmod  nmod
   pc
    nmod

   Economic news had little effect on financial markets
```
Projectivity

- Most theoretical frameworks do not assume projectivity.
- Non-projective structures are needed to account for
  - long-distance dependencies,
  - free word order.
The problem:
- Input: Sentence $x = w_0, w_1, \ldots, w_n$ with $w_0 = \text{root}$
- Output: Dependency graph $G = (V, A)$ for $x$ where:
  - $V = \{0, 1, \ldots, n\}$ is the vertex set,
  - $A$ is the arc set, i.e., $(i, j, k) \in A$ represents a dependency from $w_i$ to $w_j$ with label $l_k \in L$

Two main approaches:
- Grammar-based parsing
  - Context-free dependency grammar
  - Constraint dependency grammar
- Data-driven parsing
  - Transition-based models
  - Graph-based models
Context-Free Dependency Grammar

- Dependency grammar as lexicalized context-free grammar:
  - $H \rightarrow L_1 \cdots L_m h R_1 \cdots R_n$
  - $H \in V_N; h \in V_T; L_1 \cdots L_m, R_1 \cdots R_n \in V_N^*$

- Standard context-free parsing algorithms (CKY, Earley, etc.)

- Projective, unlabeled dependency trees only

- Weakly equivalent to (arbitrary) context-free grammars
  
  [Hays 1964, Gaifman 1965]

- Recent developments:
  - Link Grammar [Sleator and Temperley 1991]
  - Earley-style parser with left-corner filtering
    [Lombardo and Lesmo 1996]
Constraint Dependency Grammar

- Parsing as constraint satisfaction [Maruyama 1990]:
  - Grammar consists of a set of boolean constraints, i.e. logical formulas that describe well-formed dependency graphs.
  - Constraint propagation removes candidate graphs that contradict constraints (eliminative parsing).
- Handles non-projective labeled dependency graphs
- Parsing intractable in the general case
- Recent developments:
  - Weighted Constraint Dependency Grammar
    [Menzel and Schröder 1998, Foth et al. 2004]
  - Probabilistic Constraint Dependency Grammar
  - Topological/Extensible Dependency Grammar
    [Duchier and Debusmann 2001, Debusmann et al. 2004]
Transition-Based Models

- **Basic idea:**
  - Define a transition system (state machine) for mapping a sentence to its dependency graph.
  - **Learning:** Induce a model for predicting the next state transition, given the transition history.
  - **Parsing:** Construct the optimal transition sequence, given the induced model.

- **Characteristics:**
  - Local training of a model for optimal transitions
  - Greedy search/inference
Graph-Based Models

- Basic idea:
  - Define a space of candidate dependency graphs for a sentence.
  - **Learning**: Induce a model for scoring an entire dependency graph for a sentence.
  - **Parsing**: Find the highest-scoring dependency graph, given the induced model.

- Characteristics:
  - Global training of a model for optimal dependency graphs
  - Exhaustive search/inference
Pros and Cons of Dependency Parsing

What are the advantages of dependency-based methods?
What are the disadvantages?
Four types of considerations:
- Complexity
- Transparency
- Word order
- Expressivity
Complexity

- Practical complexity:
  - Given the **Single-Head** constraint, parsing a sentence $x = w_1, \ldots, w_n$ can be reduced to labeling each token $w_i$ with:
    - a head word $h_i$,
    - a dependency type $d_i$.

- Theoretical complexity:
  - By exploiting the special properties of dependency graphs, it is sometimes possible to improve worst-case complexity compared to constituency-based parsing:
    - Lexicalized parsing in $O(n^3)$ time [Eisner 1996]
Transparency

- Direct encoding of predicate-argument structure
Pros and Cons of Dependency Parsing

Transparency

- Direct encoding of predicate-argument structure
- Fragments directly interpretable

```
She writes books
```

```
NP
  | NP
  | PRP VBZ
  | She writes
  | NNS books

sbj

```

Introduction to Data-Driven Dependency Parsing 28(1)
Transparency

- Direct encoding of predicate-argument structure
- Fragments directly interpretable
- But only with labeled dependency graphs

```
sbj
     ↓
 She  writes  books
```

```
NP     
 │     
 PRP  VBZ
     │
She  writes
     ↓
 NP     
     │
 NNS  books
```
Word Order

- Dependency structure independent of word order
- Suitable for free word order languages

hon har sett honom

(she) (has) (seen) (him)

Venn diagram: S → VP → NP

(she) (has) (seen) (him)
Word Order

- Dependency structure independent of word order
- Suitable for free word order languages

```
(S)
  VP
  NP
  NP
  PRP PRP VBN
  honom har hon sett
  (him) (has) (she) (seen)
```

```
S
  VP
  NP
  NP
  PRP VB PRP VBN
  honom har hon sett
  (him) (has) (she) (seen)
```
Word Order

- Dependency structure independent of word order
- Suitable for free word order languages
- But only with non-projective dependency graphs

(him) (has) (she) (seen)
Expressivity

- Limited expressivity:
  - Every projective dependency grammar has a strongly equivalent context-free grammar, but not vice versa [Gaifman 1965].
  - Impossible to distinguish between phrase modification and head modification in unlabeled dependency structure [Mel’čuk 1988].

What about labeled non-projective dependency structures?
Conclusion

Summary

- Dependency syntax – basic concepts
- Dependency parsing – main approaches
- Pros and cons
References and Further Reading


References and Further Reading

Dependency systems and phrase-structure systems. *Information and Control*, 8:304–337.


