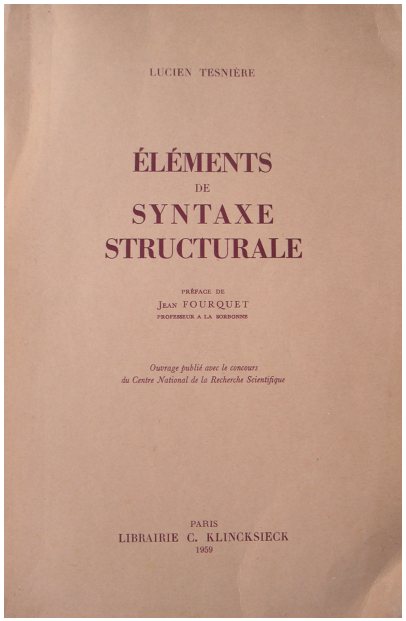
Discontinuous Dependency Parsing

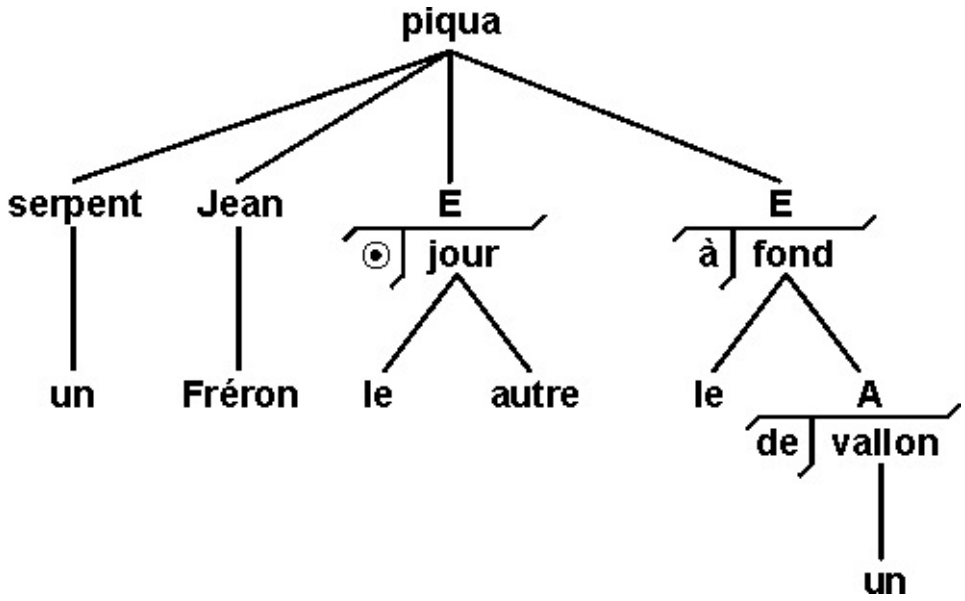
Joakim Nivre

Uppsala University
Department of Linguistics and Philology
joakim.nivre@lingfil.uu.se

50 Years Ago Today

- Lucien Tesnière: Éléments de syntaxe structurale (Paris, 1959)





Adjacency

"Le principe fondamental de la transformation de l'ordre structural en ordre linéaire est de transporter les connexions de l'ordre structural en séquences de l'ordre linéaire, de façon que les éléments qui sont en connexion dans l'ordre structurale se trouvent en voisinage immédiat sur la chaîne parlée." (Tesnière, 1959:20)

- Syntactically related elements occur next to each other:

  Economic news had little effect on financial markets.

Adjacency

"Le principe fondamental de la transformation de l'ordre structural en ordre linéaire est de transporter les connexions de l'ordre structural en séquences de l'ordre linéaire, de façon que les éléments qui sont en connexion dans l'ordre structurale se trouvent en voisinage immédiat sur la chaîne parlée." (Tesnière, 1959:20)

- Syntactically related elements occur next to each other:

  [Economic news] had [little effect on financial markets]

Adjacency

"Le principe fondamental de la transformation de l'ordre structural en ordre linéaire est de transporter les connexions de l'ordre structural en séquences de l'ordre linéaire, de façon que les éléments qui sont en connexion dans l'ordre structurale se trouvent en voisinage immédiat sur la chaîne parlée." (Tesnière, 1959:20)

- Syntactically related elements occur next to each other:

  [Economic] news had [little] effect [on financial markets]

Adjacency

"Le principe fondamental de la transformation de l'ordre structural en ordre linéaire est de transporter les connexions de l'ordre structural en séquences de l'ordre linéaire, de façon que les éléments qui sont en connexion dans l'ordre structurale se trouvent en voisinage immédiat sur la chaîne parlée." (Tesnière, 1959:20)

- Syntactically related elements occur next to each other:

  Economic news had little effect on [financial markets]

Adjacency

"Le principe fondamental de la transformation de l'ordre structural en ordre linéaire est de transporter les connexions de l'ordre structural en séquences de l'ordre linéaire, de façon que les éléments qui sont en connexion dans l'ordre structurale se trouvent en voisinage immédiat sur la chaîne parlée." (Tesnière, 1959:20)

- Syntactically related elements occur next to each other:

  Economic news had little effect on [financial] markets

Adjacency

"Le principe fondamental de la transformation de l'ordre structural en ordre linéaire est de transporter les connexions de l'ordre structural en séquences de l'ordre linéaire, de façon que les éléments qui sont en connexion dans l'ordre structurale se trouvent en voisinage immédiat sur la chaîne parlée." (Tesnière, 1959:20)

- Syntactically related elements occur next to each other:

  Economic news had little effect on financial markets.

- Modifiers of the same head may intervene:

  Economic news gave the market a shock.

Adjacency

"Le principe fondamental de la transformation de l'ordre structural en ordre linéaire est de transporter les connexions de l'ordre structural en séquences de l'ordre linéaire, de façon que les éléments qui sont en connexion dans l'ordre structurale se trouvent en voisinage immédiat sur la chaîne parlée." (Tesnière, 1959:20)

- Syntactically related elements occur next to each other:

  Economic news had little effect on financial markets.

- Modifiers of the same head may intervene:

  [Economic news] gave [the market][a shock]

Discontinuity

- But exceptions occur regularly:

  What did economic news have an effect on?

  A hearing is scheduled on the issue today.

  They had such a momentum that the wall collapsed.

Discontinuity

- But exceptions occur regularly:

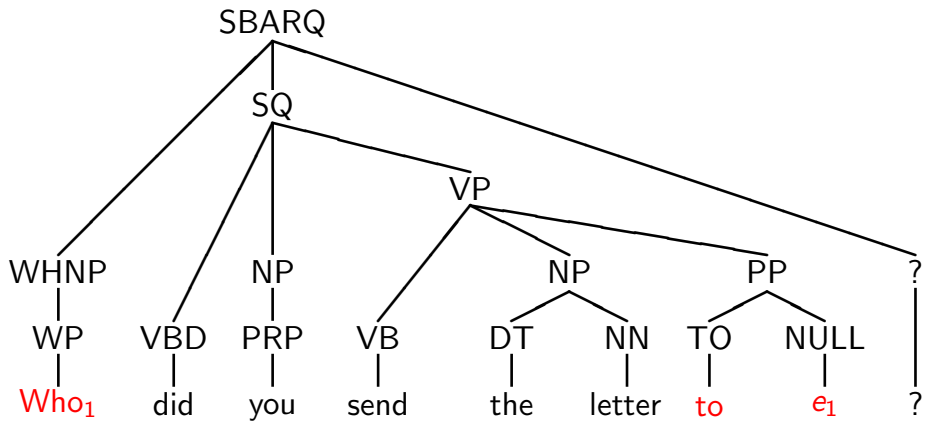
  [What] did economic news have an effect on?

  A hearing is scheduled [on the issue] today.

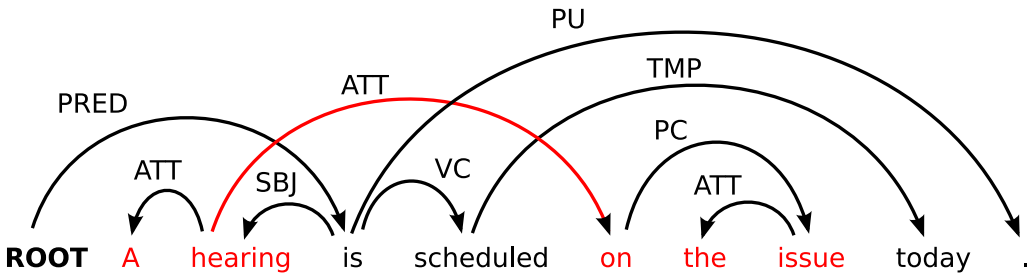
  They had such a momentum [that the wall collapsed]

Linguistic Representations

- Phrase structure trees with empty nodes:



- Non-projective dependency trees:



Natural Language Parsing

- Phrase structure parsing:
  - Empty nodes make parsing harder
  - Few (statistical) parsers can handle empty nodes
  - Low accuracy for unbounded dependencies [Rimell et al. 2009]
- Dependency parsing:
  - Non-projective trees make parsing harder
  - Many parsers only admit projective trees
  - Non-projective relations needed for representational adequacy
- This talk:
  - Parsing with non-projective dependency trees

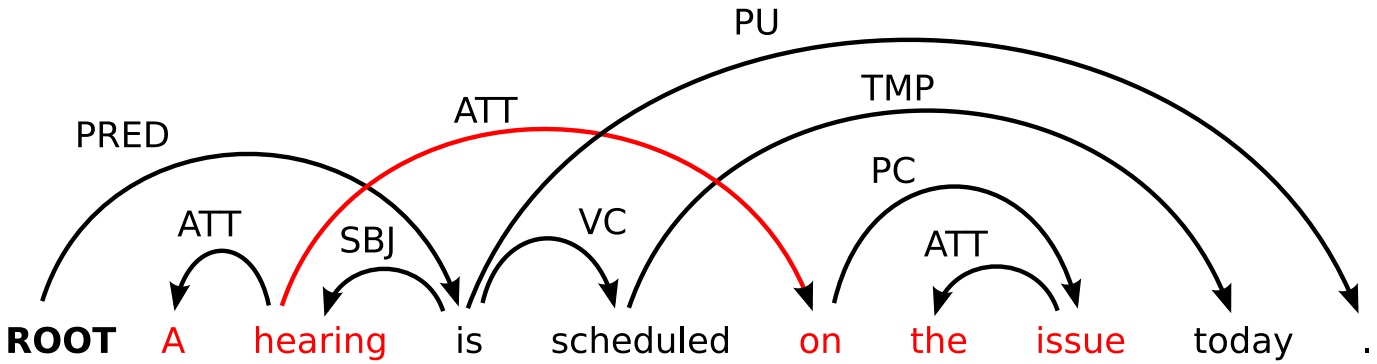
Outline

- The problem
  - Parsing with non-projective dependency trees
- Some proposed solutions:
  - Projective parsing with post-processing
  - Dynamic programming
  - Constraint satisfaction
  - Transition-based approaches
- Online reordering:
  - Reducing discontinuity to adjacency

The Problem

Dependency Trees

- A dependency tree is a labeled directed tree T, consisting of
  - a set V of nodes, labeled with words (including ROOT)
  - a set A of arcs, labeled with dependency types
  - a linear precedence order $<$ on V
- Notation:
  - Arc (w_i, w_j, l) connects head w_i to dependent w_j with label l
  - Node w_0 (labeled ROOT) is the unique root of the tree
- A dependency tree T is projective iff every subtree has a contiguous yield (every arc connects two adjacent subtrees)



A Hard Problem?

- Intractability results for parsing with non-projective trees:
  - Grammars with ID/LP separation [Neuhaus and Bröker 1997]
  - Weighted constraint grammars [Buch-Kromann 2006]
  - Spanning tree parsing with horizontal or vertical markovization or valency [McDonald and Pereira 2006, McDonald and Satta 2007]
- Complexity of deterministic parsing [Nivre 2008a]:
  - $O(n)$ with projective trees, $O(n^2)$ with arbitrary trees
- Source of difficulty:
  - Linear order unconstrained by tree structure (all permutations)
  - Trees composed of non-adjacent subtrees (discontinuity)

A Significant Problem?

Language	Sentences	Dependencies
Arabic [Maamouri and Bies 2004]	11.2%	0.4%
Basque [Aduriz et al. 2003]	26.2%	2.9%
Czech [Hajič et al. 2001]	23.2%	1.9%
Danish [Kromann 2003]	15.6%	1.0%
Greek [Prokopidis et al. 2005]	20.3%	1.1%
Russian [Boguslavsky et al. 2000]	10.6%	0.9%
Slovene [Džeroski et al. 2006]	22.2%	1.9%
Turkish [Oflazer et al. 2003]	11.6%	1.5%

A Middle Ground?

- Mildly non-projective dependency trees:
  - Trees with gap degree at most k [Holan et al. 1998]
  - Trees with arc degree at most k [Nivre 2006]
  - Trees that are k-planar [Yli-Jyrä 2003]
  - Trees that are well-nested [Bodirsky et al. 2005]
- Important result [Kuhlmann and Möhl 2007]:
  - Well-nested with gap degree 1 $\Leftrightarrow$ mildly context-sensitive
- Good coverage in treebanks [Kuhlmann and Nivre 2006, Havelka 2007]

Constraint	Czech	Danish
Gap degree at most 1	99.6%	99.8%
Arc degree at most 1	99.5%	98.2%
Well-nested	99.5%	99.2%

Some Proposed Solutions

Projective Parsing with Post-Processing

- Pure post-processing:
  - Corrective modeling [Hall and Novák 2005]
- Post-processing combined with machine learning:
  - Pseudo-projective parsing [Nivre and Nilsson 2005]
  - Approximate spanning tree parsing [McDonald and Pereira 2006]
- Results for Czech (unlabeled attachment score):

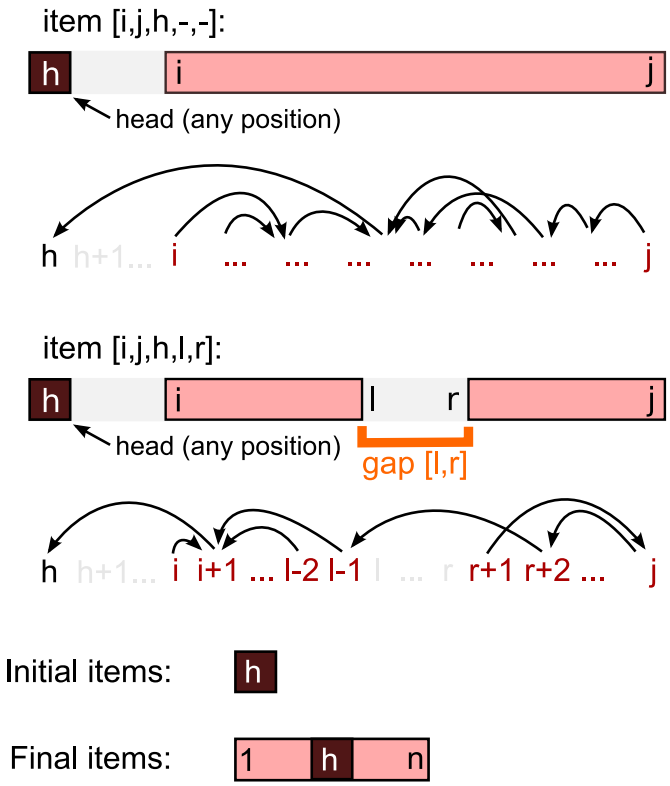
Method	**Before**	**After**
Corrective modeling	84.3%	85.0%
Pseudo-projective parsing	78.5%	80.1%
Approximate spanning tree parsing	84.2%	85.2%

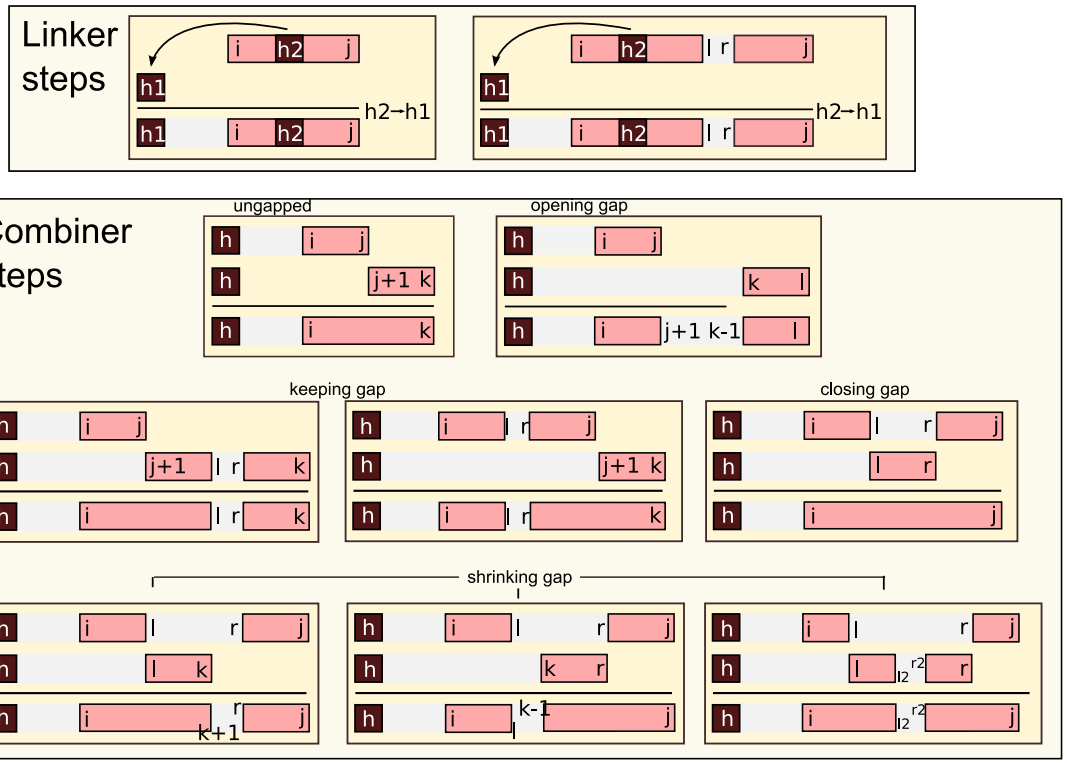
Dynamic Programming

- Dynamic programming widely used for projective parsing:
  - Link Grammar parsing [Sleator and Temperley 1993]
  - Earley style parsing [Lombardo and Lesmo 1998]
  - CKY style bilexical parsing [Collins 1996]
  - Split-head bilexical parsing [Eisner 1996, Eisner and Satta 1999]
- Standard algorithms impose the adjacency constraint:
  - $[w_i, \ldots, w_k] + [w_{k+1}, \ldots, w_j] \Rightarrow [w_i, \ldots, w_j]$
- Extensions to mildly non-projective dependency trees:
  - Well-nested with gap degree 1 [Kuhlmann and Satta 2009]
    - $O(n^6)$ for unlexicalized parsing, $O(n^8)$ for lexicalized parsing
  - Well-nested with gap degree k [Gómez-Rodríguez et al. 2009]
    - $O(n^{5+2k})$ for lexicalized parsing using the hook trick

Well-Nested with Gap Degree 1

[Gómez-Rodríguez et al. 2009]





Constraint Satisfaction

- Dependency parsing as constraint satisfaction [Maruyama 1990]:
  - Variables $h_1, \ldots, h_n$ with domain $V = \{w_0, \ldots, w_n\}$
  - Grammar G = Set of boolean constraints C
  - Parsing = Search for a tree in $\{T \mid \forall C \in G : C(T)\}$
  - Projectivity is just another constraint
- Adding soft weighted constraints [Menzel and Schröder 1998]:

$$\arg\max_T \mathbf{w}(T) = \arg\max_T \prod_{C:\neg C(T)} \mathbf{w}(C)$$

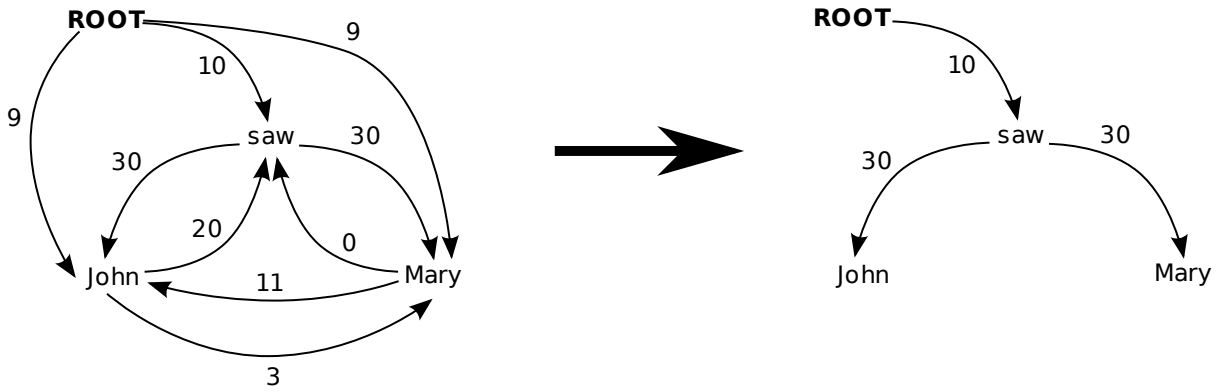
- Main problem: How perform search efficiently?
  - WCDG uses transformational search [Foth et al. 2004]

Arc-Factored Models

- Arc-factored spanning tree parsing [McDonald et al. 2005]:
  - Constraint $C_{(w_i,w_j,l)}(T)$ is **true** if $(w_i, w_j, l) \in T$
  - Maximum directed spanning tree in complete graph:

$$\arg\max_T \mathbf{w}(T) = \arg\max_T \sum_{C:C(T)} \mathbf{w}(C)$$

  - Exact parsing in $O(n^2)$ time [Chu and Liu 1965, Edmonds 1967]
  - Improvement for Czech over projective model: 83.3% $\rightarrow$ 84.4%



Beyond Arc-Factored Models

- Going beyond single arcs makes parsing intractable:
  - Horizontal markovization [McDonald and Pereira 2006]
  - Vertical markovization [McDonald and Satta 2007]
  - Valency [McDonald and Satta 2007]

"An island of tractability in a swamp of NP-completeness." (David Smith)

- Related approaches using approximate inference:
  - Gibbs sampling [Nakagawa 2007]
  - Loopy belief propagation [Smith and Eisner 2008]
  - Integer linear programming [Riedel et al. 2006, Martins et al. 2009]

Transition-Based Approaches

- Transition-based dependency parsing:
  - Define a transition system for dependency parsing
  - Train a classifier for predicting the next transition
  - Use the classifier to do parsing as greedy, deterministic search
- Complexity:
  - The worst-case time complexity of a deterministic parser is given by an upper bound on the length of transition sequences

Projective Shift-Reduce Parsing

Configuration: $(Stack, Buffer, Arcs)$

Initial: $([w_0]_S, [w_1, \ldots, w_n]_B, \{\ \}_A)$

Terminal: $([w_0]_S, [\ ]_B, A)$

Shift:

$([\ldots, w_i]_S, [w_j, \ldots]_B, A) \Rightarrow ([\ldots, w_i, w_j]_S, [\ldots]_B, A)$

Right-Arc(l):

$([\ldots, w_i, w_j]_S, B, A) \Rightarrow ([\ldots, w_i]_S, B, A \cup \{(w_i, w_j, l)\})$

Left-Arc(l) [$i \neq 0$]:

$([\ldots, w_i, w_j]_S, B, A) \Rightarrow ([\ldots, w_j]_S, B, A \cup \{(w_j, w_i, l)\})$

From Projective to Non-Projective Trees

- Deterministic projective parsing runs in $O(n)$ time [Nivre 2003]
- Adding arcs between non-adjacent subtrees [Attardi 2006]:
  - RA′: $([\ldots, w_i, w_j, w_k]_S, B, A) \Rightarrow ([\ldots, w_i, w_j]_S, B, A \cup \{(w_i, w_k, l)\})$
  - LA′: $([\ldots, w_i, w_j, w_k]_S, B, A) \Rightarrow ([\ldots, w_j, w_k]_S, B, A \cup \{(w_k, w_i, l)\})$

  Maintains $O(n)$ time for a subset of non-projective trees
- Descending through the stack [Covington 2001, Nivre 2007]:
  - After shifting w_i, search S from the top for possible links
  - Do not remove dependents from S when adding arcs

  Runs in $O(n^2)$ time with arbitrary non-projective trees
  - Improvement for Czech over projective model: 84.5% → 84.9%
  - Parsing time can be reduced by limiting arc degree [Nivre 2007]

Summing Up

- Post-processing:
  - Most post-processing methods run in $O(n^2)$ time or better
  - Overall complexity also depends on base parser
- Dynamic programming:
  - Polynomial time parsing for subsets of non-projective trees
- Constraint satisfaction:
  - Parsing in $O(n^2)$ time with arc-factored model
  - Parsing problem intractable for more complex models
- Transition-based parsing:
  - Deterministic parsing in $O(n^2)$ time ($O(n)$ for subsets)

Online Reordering

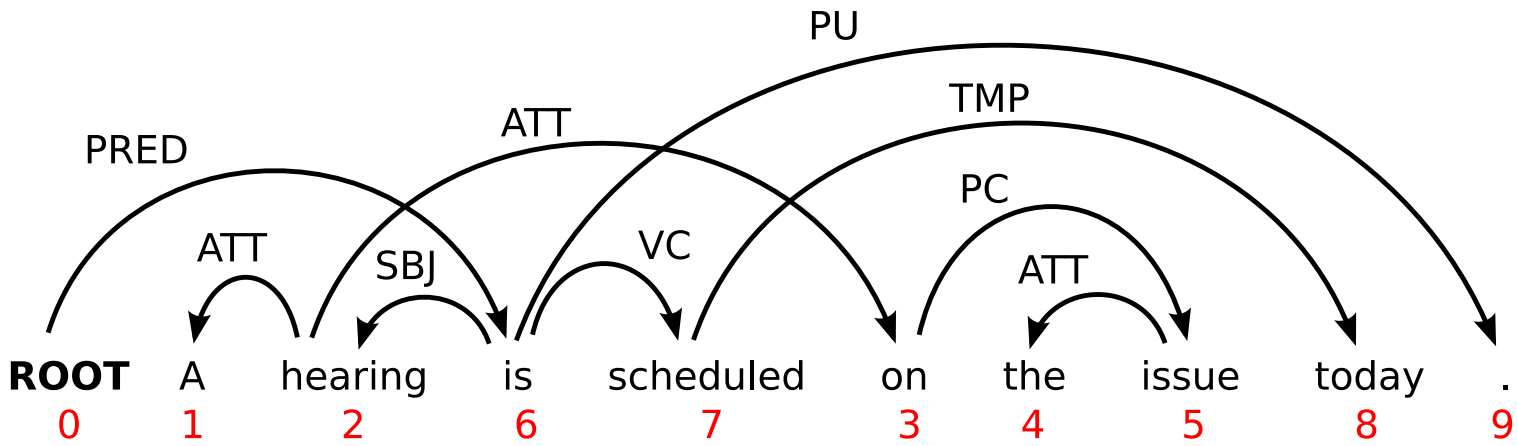
A New Tack?

- Two basic strategies:
  - Relax adjacency constraints to permit discontinuity:
    - Dynamic programming with gaps
    - Transition-based parsing with non-adjacent link operations
  - Reduce discontinuity and adjacency to a single common case:
    - Constraint satisfaction – ignore word order
    - Post-processing – project to different structure
- New twist on second strategy:
  - Reduce discontinuity to adjacency by reordering input words

    [Nivre 2008b, Titov et al. 2009, Nivre 2009]

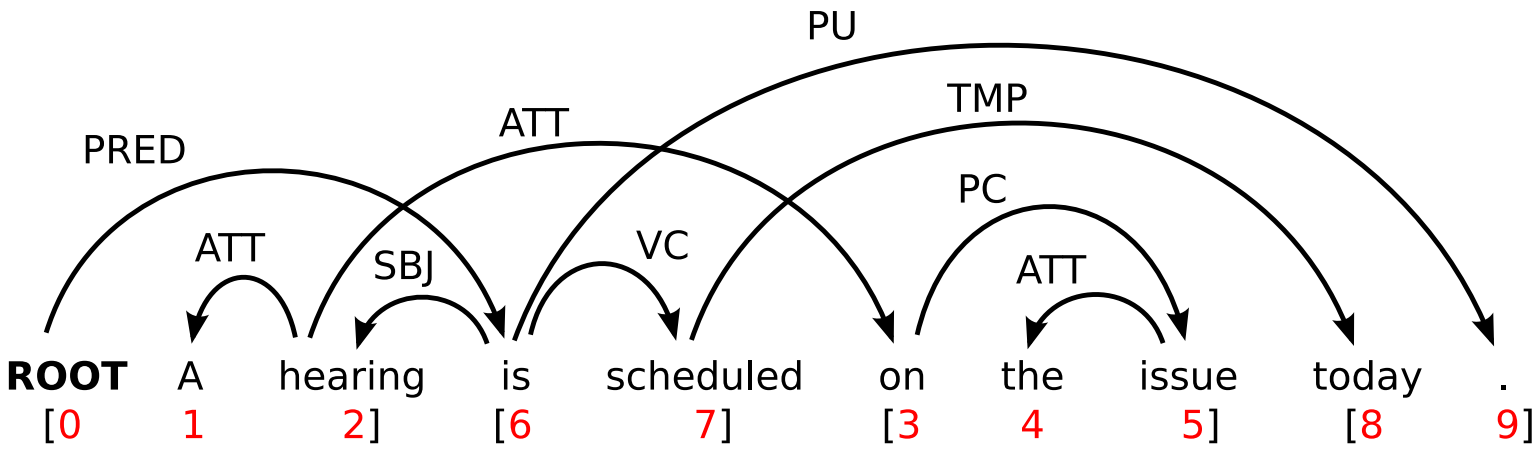
Projectivity and Word Order

- Projectivity is a property of a dependency tree only in relation to a particular word order
- Words can always be reordered to make the tree projective
- Given a dependency tree $T = (V, A, <)$, let the projective order $<_p$ be the order defined by an inorder traversal of T with respect to $<$ [Veselá et al. 2004]



Projectivity and Word Order

- Projectivity is a property of a dependency tree only in relation to a particular word order
- Words can always be reordered to make the tree projective
- Given a dependency tree $T = (V, A, <)$, let the projective order $<_p$ be the order defined by an inorder traversal of T with respect to $<$ [Veselá et al. 2004]



Parsing with Online Reordering

- Transition-based parsing with two interleaved processes:
  - Sort words into the projective order $<_p$
  - Build dependency tree T by connecting adjacent subtrees
    - T is always projective with respect to $<_p$
    - T may be non-projective with respect to $<$
- Simplest strategy:
  - Keep S sorted with respect to $<_p$
  - Always extract from B the least element with respect to $<_p$
  - Projective parsing as a special case:
    - Shift = Extract-Least(B, $<_p$) because $< \Leftrightarrow <_p$

Shift-Reduce Parsing with Online Reordering

Configuration: $(Stack, Buffer, Arcs)$

Initial: $([w_0]_S, [w_1, \ldots, w_n]_B, \{\ \}_A)$

Terminal: $([w_0]_S, [\ ]_B, A)$

Shift:

$$([\ldots, w_i]_S, [w_j, \ldots]_B, A) \Rightarrow ([\ldots, w_i, w_j]_S, [\ldots]_B, A)$$

Right-Arc(l):

$$([\ldots, w_i, w_j]_S, B, A) \Rightarrow ([\ldots, w_i]_S, B, A \cup \{(w_i, w_j, l)\})$$

Left-Arc(l) [$i \neq 0$]:

$$([\ldots, w_i, w_j]_S, B, A) \Rightarrow ([\ldots, w_j]_S, B, A \cup \{(w_j, w_i, l)\})$$

Swap [$0 < i < j$]:

$$([\ldots, w_i, w_j]_S, [\ldots]_B, A) \Rightarrow ([\ldots, w_j]_S, [w_i, \ldots]_B, A)$$

Parsing Example

$[\text{ROOT}_{0(0)}]_S \quad [\text{A}_{1(1)}, \text{hearing}_{2(2)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{A}_{1(1)}]_S \quad [\text{hearing}_{2(2)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{A}_{1(1)}, \text{hearing}_{2(2)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{is}_{3(6)}]_S \quad [\text{scheduled}_{4(7)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}]_S \quad [\text{on}_{5(3)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{on}_{5(3)}]_S \quad [\text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{is}_{3(6)}, \text{on}_{5(3)}]_S \quad [\text{scheduled}_{4(7)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{is}_{3(6)}]_S \quad [\text{scheduled}_{4(7)}, \text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}]_S \quad [\text{the}_{6(4)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{the}_{6(4)}]_S \quad [\text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{is}_{3(6)}, \text{the}_{6(4)}]_S \quad [\text{scheduled}_{4(7)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{the}_{6(4)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{is}_{3(6)}]_S \quad [\text{scheduled}_{4(7)}, \text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}]_S \quad [\text{issue}_{7(5)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{issue}_{7(5)}]_S \quad [\text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{is}_{3(6)}, \text{issue}_{7(5)}]_S \quad [\text{scheduled}_{4(7)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{the}_{6(4)}, \text{issue}_{7(5)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$A_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}, \text{issue}_{7(5)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$\text{A}_1 \xleftarrow{\text{ATT}} \text{hearing}_2$$
$$\text{the}_6 \xleftarrow{\text{ATT}} \text{issue}_7$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{on}_{5(3)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{today}_{8(8)}, \cdot_{9(9)}]_B$

$$\begin{aligned} A_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\ \text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\ \text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \end{aligned}$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}]_S \quad [\text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{today}_{8(8)}, .{}_{9(9)}]_B$

$$\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5
\end{aligned}$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{hearing}_{2(2)}, \text{is}_{3(6)}]_S \quad [\text{scheduled}_{4(7)}, \text{today}_{8(8)}, .9_{(9)}]_B$

$$\begin{aligned}
A_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5
\end{aligned}$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{is}_{3(6)}]_S \quad [\text{scheduled}_{4(7)}, \text{today}_{8(8)}, .9_{(9)}]_B$

$$\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3
\end{aligned}$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}]_S \quad [\text{today}_{8(8)}, .{}_{9(9)}]_B$

$$\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3
\end{aligned}$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}, \text{today}_{8(8)}]_S \quad [.{}_{9(9)}]_B$

$$\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3
\end{aligned}$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{is}_{3(6)}, \text{scheduled}_{4(7)}]_S \quad [\cdot_{9(9)}]_B$

$$
\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3 \\
\text{scheduled}_4 &\xrightarrow{\text{TMP}} \text{today}_8
\end{aligned}
$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{is}_{3(6)}]_S \quad [\cdot_{9(9)}]_B$

$$
\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3 \\
\text{scheduled}_4 &\xrightarrow{\text{TMP}} \text{today}_8 \\
\text{is}_3 &\xrightarrow{\text{VC}} \text{scheduled}_4
\end{aligned}
$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{is}_{3(6)}, .{}_{9(9)}]_S \quad [\,]_B$

$$
\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3 \\
\text{scheduled}_4 &\xrightarrow{\text{TMP}} \text{today}_8 \\
\text{is}_3 &\xrightarrow{\text{VC}} \text{scheduled}_4
\end{aligned}
$$

Parsing Example

$[\text{ROOT}_{0(0)}, \text{is}_{3(6)}]_S \quad [\,]_B$

$$
\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3 \\
\text{scheduled}_4 &\xrightarrow{\text{TMP}} \text{today}_8 \\
\text{is}_3 &\xrightarrow{\text{VC}} \text{scheduled}_4 \\
\text{is}_3 &\xrightarrow{\text{PU}} .9
\end{aligned}
$$

Parsing Example

$[\text{ROOT}_{0(0)}]_S \quad [\,]_B$

$$
\begin{aligned}
\text{A}_1 &\xleftarrow{\text{ATT}} \text{hearing}_2 \\
\text{the}_6 &\xleftarrow{\text{ATT}} \text{issue}_7 \\
\text{on}_5 &\xrightarrow{\text{PC}} \text{issue}_7 \\
\text{hearing}_2 &\xrightarrow{\text{ATT}} \text{on}_5 \\
\text{hearing}_2 &\xleftarrow{\text{SBJ}} \text{is}_3 \\
\text{scheduled}_4 &\xrightarrow{\text{TMP}} \text{today}_8 \\
\text{is}_3 &\xrightarrow{\text{VC}} \text{scheduled}_4 \\
\text{is}_3 &\xrightarrow{\text{PU}} .9 \\
\text{ROOT}_0 &\xrightarrow{\text{PRED}} \text{is}_3
\end{aligned}
$$

Training Oracle

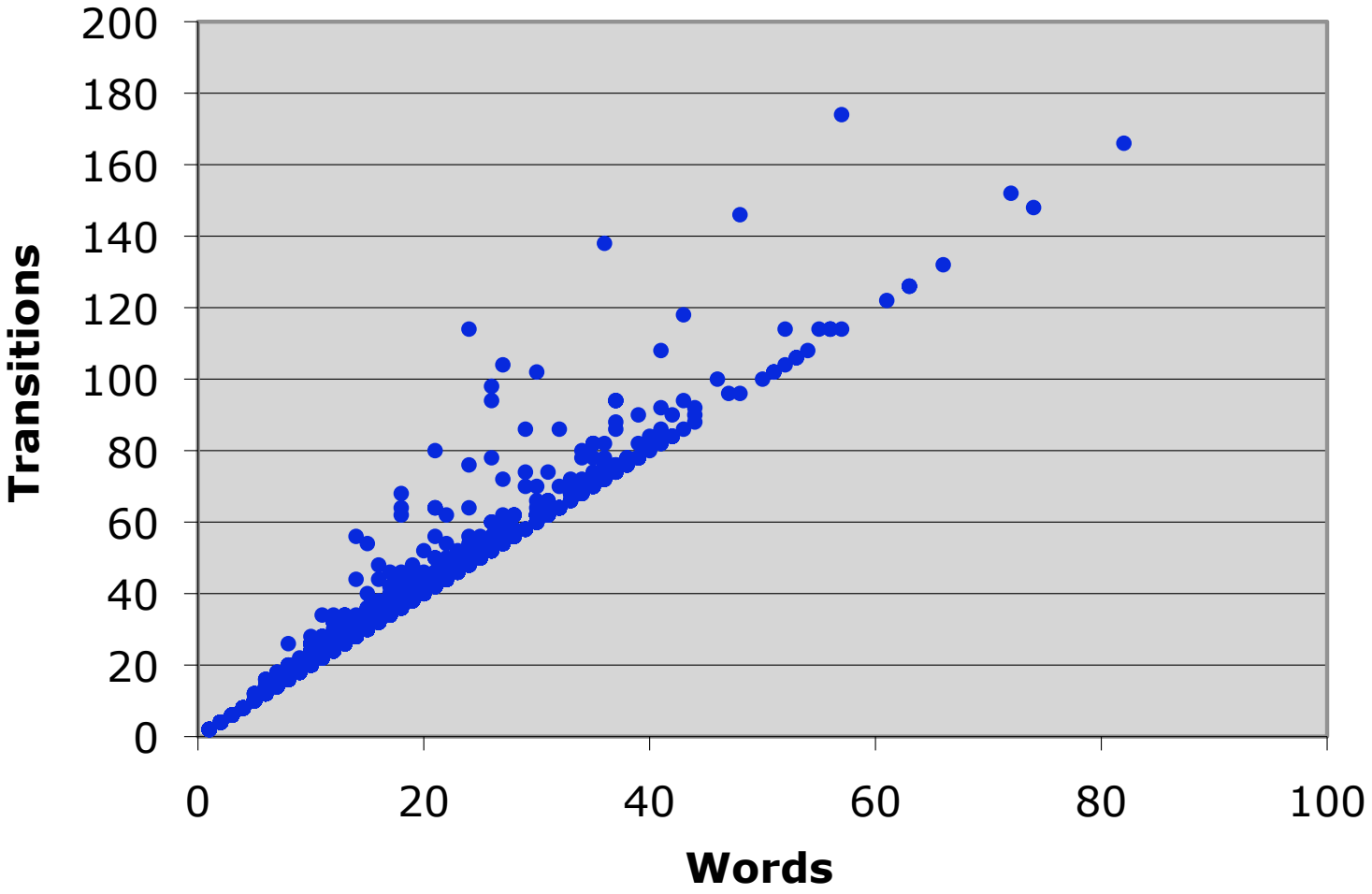
- To train classifiers we need an oracle to answer the question:
  - What sequence of transitions derives a given tree T?
- Oracle for keeping S sorted (eager swapping):
  - In a configuration $c = ([\ldots, w_i, w_j]_S, B, A)$, do

Left-Arc$_l$	if $w_i \overset{l}{\leftarrow} w_j$ and w_i complete
Right-Arc$_l$	if $w_i \overset{l}{\rightarrow} w_j$ and w_j complete
Swap	if $w_i >_p w_j$
Shift	otherwise

Algorithm Analysis

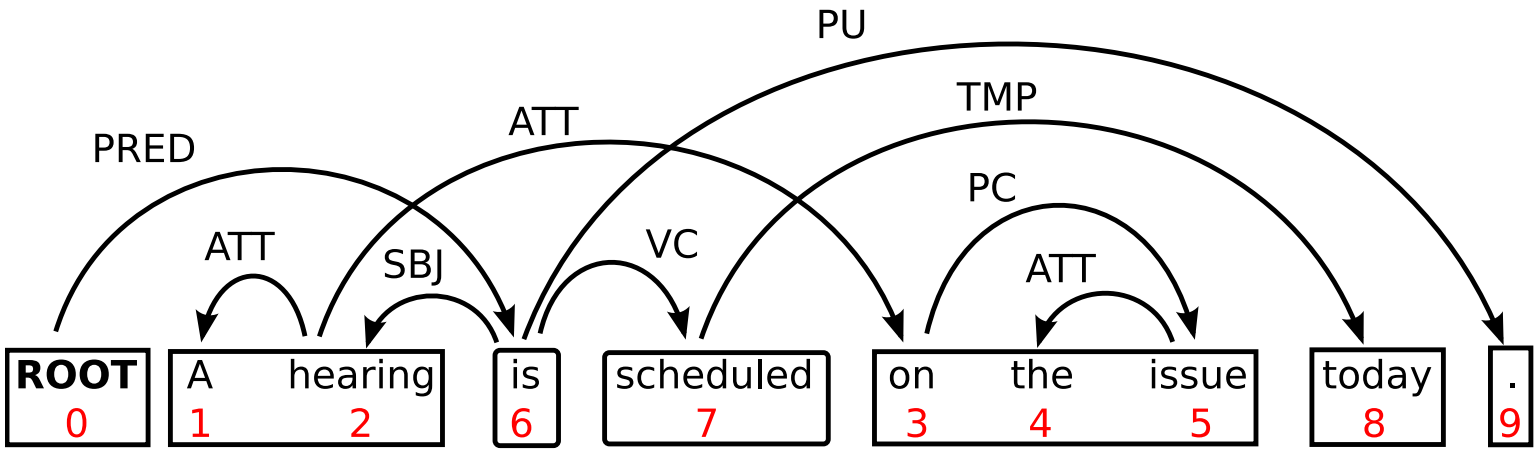
- Correctness:
  - Sound and complete for arbitrary dependency trees
- Time complexity of deterministic parsing:
  - Parser terminates after $2n + 2k$ transitions ($k = \#\text{swaps}$)
  - $O(n^2)$ in the worst case ($k = \frac{n(n-1)}{2}$)
  - $O(n)$ in the best case ($k = 0$)
  - Average case $\approx$ best case?
- Questions:
  - Can we train classifiers to do reordering as well as parsing?
  - Can we maintain linear parsing time on average?

Abstract Running Time – Czech



Improving the Oracle

- Problem:
  - Keeping S sorted results in more swaps than necessary
  - Potentially negative for both efficiency and accuracy
- Improved strategy:
  - Build maximal projective components before swapping
  - MPCs = connected components of oracle parse without swaps



New Oracle

- Oracle for building MPCs (lazy swapping):
  - In a configuration $c = ([\ldots, w_i, w_j], B, A)$, do

Left-Arc$_l$	if $w_i \overset{l}{\leftarrow} w_j$ and w_i complete
Right-Arc$_l$	if $w_i \overset{l}{\rightarrow} w_j$ and w_j complete
Swap	if $w_i >_p w_j$, $B = [w_k, \ldots]$ and $\text{mpc}(w_j) \neq \text{mpc}(w_k)$
Shift	otherwise

- Reduces number of swaps significantly:

Oracle	Arabic	Czech	Danish	Slovene	Turkish
Old	1416	57011	8296	2191	2828
New	229	26208	1497	690	1253

Parsing Accuracy – Czech

Parser	LAS	Prec	Rec	LEM
Online reordering	82.7	79.3	71.0	35.3
MaltParser [Nivre et al. 2006]	78.4	76.3	57.3	27.4
MSTParser [McDonald et al. 2006]	80.2	60.5	61.7	29.9
MST_{Malt} [Nivre and McDonald 2008]	82.3	63.9	69.2	31.2

LAS = Labeled attachment score
Prec/Rec = Precision/Recall on non-projective arcs
LEM = Labeled Exact Match

- Improved accuracy on non-projective arcs:
  - Better precision than approximate MST parsers
  - Better recall than pseudo-projective parsing (Malt)
  - Larger improvement for LEM than for LAS

Parsing Accuracy – Czech

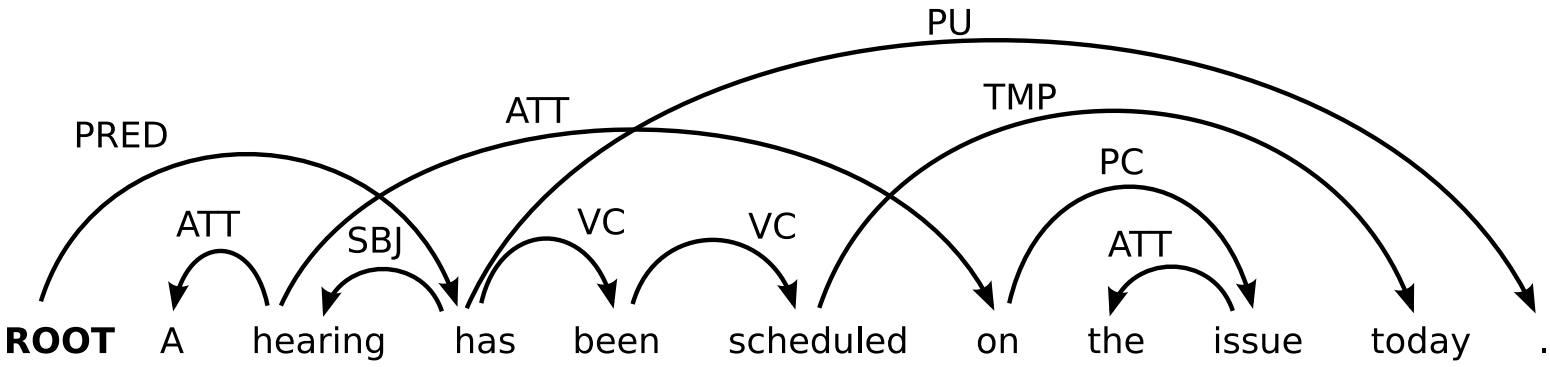
Parser	LAS	Prec	Rec	LEM
Online reordering	82.7	79.3	71.0	35.3
MaltParser [Nivre et al. 2006]	78.4	76.3	57.3	27.4
MSTParser [McDonald et al. 2006]	80.2	60.5	61.7	29.9
MST_{Malt} [Nivre and McDonald 2008]	82.3	63.9	69.2	31.2

LAS = Labeled attachment score
Prec/Rec = Precision/Recall on non-projective arcs
LEM = Labeled Exact Match

- Improved accuracy on non-projective arcs:
  - Better precision than approximate MST parsers
  - Better recall than pseudo-projective parsing (Malt)
  - Larger improvement for LEM than for LAS
- Try it out yourself: http://maltparser.org

An Open Problem

- Yet another class of mildly non-projective trees:
  - Trees that can be parsed with at most k swaps
- Problem:
  - Determine minimum number of swaps for an arbitrary tree
- Partial result:
  - The MPC oracle does not give the minimum number of swaps
  - Proof by counterexample



Conclusion

Conclusion

- Parsing discontinuous constructions:
  - Necessary for representational adequacy
  - Computationally challenging
- Variety of approaches:
  - Dynamic programming
  - Constraint satisfaction
  - Transition-based approaches
  - Projective parsing with post-processing
- Empirical results:
  - Minor improvements at high computational costs?
  - Standard evaluation metrics are not always informative

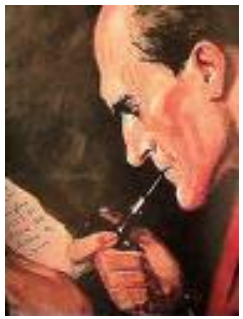
The Way Forward

- Deeper empirical analysis:
  - Systematic comparison of different methods
  - More fine-grained evaluation metrics
  - Analysis of specific linguistic constructions
- Data-driven parsing research:
  - Linguistic data for training and evaluation
  - Empirical analysis to guide theory development

The Way Forward

- Deeper empirical analysis:
  - Systematic comparison of different methods
  - More fine-grained evaluation metrics
  - Analysis of specific linguistic constructions
- Data-driven parsing research:
  - Linguistic data for training and evaluation
  - Empirical analysis to guide theory development

"The temptation to form premature theories upon insufficient data is the bane of our profession." (Sherlock Holmes)



- I. Aduriz, M. J. Aranzabe, J. M. Arriola, A. Atutxa, A. Díaz de Ilarraza, A. Garmendia, and M. Oronoz. 2003. Construction of a Basque dependency treebank. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 201–204.
- Giuseppe Attardi. 2006. Experiments with a multilanguage non-projective dependency parser. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 166–170.
- Manuel Bodirsky, Marco Kuhlmann, and Mattias Möhl. 2005. Well-nested drawings as models of syntactic structure. In *10th Conference on Formal Grammar and 9th Meeting on Mathematics of Language*.
- Igor Boguslavsky, Svetlana Grigorieva, Nikolai Grigoriev, Leonid Kreidlin, and Nadezhda Frid. 2000. Dependency treebank for Russian: Concept, tools, types of information. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING)*, pages 987–991.
- Matthias Buch-Kromann. 2006. *Discontinuous Grammar: A Model of Human Parsing and Language Acquisition*. Ph.D. thesis, Copenhagen Business School.
- Y. J. Chu and T. H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400.
- Michael Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 184–191.
- Michael A. Covington. 2001. A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102.
- Sašo Džeroski, Tomaž Erjavec, Nina Ledinek, Petr Pajas, Zdenek Žabokrtsky, and Andreja Žele. 2006. Towards a Slovene dependency treebank. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*.
- Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71B:233–240.
- Jason Eisner and Giorgio Satta. 1999. Efficent parsing for bilexical context-free grammars and head automaton grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 457–464.

- Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 340–345.
- Kilian Foth, Michael Daum, and Wolfgang Menzel. 2004. A broad-coverage parser for German based on defeasible constraints. In *Proceedings of KONVENS 2004*, pages 45–52.
- Carlos Gómez-Rodríguez, David Weir, and John Carroll. 2009. Parsing mildly non-projective dependency structures. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 291–299.
- Jan Hajič, Barbora Vidova Hladka, Jarmila Panevová, Eva Hajičová, Petr Sgall, and Petr Pajas. 2001. Prague Dependency Treebank 1.0. LDC, 2001T10.
- Keith Hall and Vaclav Novák. 2005. Corrective modeling for non-projective dependency parsing. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT)*, pages 42–52.
- Jiri Havelka. 2007. Beyond projectivity: Multilingual evaluation of constraints and measures on non-projective structures. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 608–615.
- Tomáš Holan, Vladislav Kuboň, Karel Oliva, and Martin Plátek. 1998. Unit coordination and gapping in dependency theory. In *Proceedings of the Workshop on Processing of Dependency-Based Grammars (ACL-COLING)*, pages 11–20.
- Matthias Trautner Kromann. 2003. The Danish Dependency Treebank and the DTAG treebank tool. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 217–220.
- Marco Kuhlmann and Mathias Möhl. 2007. Mildly context-sensitive dependency languages. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 160–167.
- Marco Kuhlmann and Joakim Nivre. 2006. Mildly non-projective dependency structures. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 507–514.

- Marco Kuhlmann and Giorgio Satta. 2009. Treebank grammar techniques for non-projective dependency parsing. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 478–486.
- Vincenzo Lombardo and Leonardo Lesmo. 1998. Unit coordination and gapping in dependency theory. In *Proceedings of the Workshop on Processing of Dependency-Based Grammars (ACL-COLING)*, pages 11–20.
- Mohamed Maamouri and Ann Bies. 2004. Developing an Arabic treebank: Methods, guidelines, procedures, and tools. In *Proceedings of the Workshop on Computational Approaches to Arabic Script-Based Languages*, pages 2–9.
- Andre Martins, Noah Smith, and Eric Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 342–350.
- Hiroshi Maruyama. 1990. Structural disambiguation with constraint propagation. In *Proceedings of the 28th Meeting of the Association for Computational Linguistics (ACL)*, pages 31–38.
- Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 81–88.
- Ryan McDonald and Giorgio Satta. 2007. On the complexity of non-projective data-driven dependency parsing. In *Proceedings of the 10th International Conference on Parsing Technologies (IWPT)*, pages 122–131.
- Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 523–530.
- Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 216–220.

- Wolfgang Menzel and Ingo Schröder. 1998. Decision procedures for dependency parsing using graded constraints. In *Proceedings of the Workshop on Processing of Dependency-Based Grammars (ACL-COLING)*, pages 78–87.
- Tetsuji Nakagawa. 2007. Multilingual dependency parsing using global features. In *Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007*, pages 952–956.
- Peter Neuhaus and Norbert Bröker. 1997. The complexity of recognition of linguistically adequate dependency grammars. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL) and the 8th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 337–343.
- Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 950–958.
- Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 99–106.
- Joakim Nivre, Johan Hall, Jens Nilsson, Gülsen Eryiğit, and Svetoslav Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 221–225.
- Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.
- Joakim Nivre. 2006. Constraints on non-projective dependency graphs. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 73–80.
- Joakim Nivre. 2007. Incremental non-projective dependency parsing. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT)*, pages 396–403.
- Joakim Nivre. 2008a. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34:513–553.

- Joakim Nivre. 2008b. Sorting out dependency parsing. In *Proceedings of the 6th International Conference on Natural Language Processing (GoTAL)*, pages 16–27.
- Joakim Nivre. 2009. Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 351–359.
- Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür. 2003. Building a Turkish treebank. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 261–277. Kluwer.
- P. Prokopidis, E. Desypri, M. Koutsombogera, H. Papageorgiou, and S. Piperidis. 2005. Theoretical and practical issues in the construction of a Greek dependency treebank. In *Proceedings of the 3rd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 149–160.
- Sebastian Riedel, Ruket Çakıcı, and Ivan Meza-Ruiz. 2006. Multi-lingual dependency parsing with incremental integer linear programming. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 226–230.
- Laura Rimell, Stephen Clark, and Mark Steedman. 2009. Unbounded dependency recovery for parser evaluation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 813–821.
- Daniel Sleator and Davy Temperley. 1993. Parsing English with a link grammar. In *Proceedings of the Third International Workshop on Parsing Technologies (IWPT)*, pages 277–292.
- David Smith and Jason Eisner. 2008. Dependency parsing by belief propagation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 145–156.
- Ivan Titov, James Henderson, Paola Merlo, and Gabriele Musillo. 2009. Online graph planarization for synchronous parsing of semantic and syntactic dependencies. In *International Joint Conferences on Artificial Intelligence (IJCAI-09)*.
- Katerina Veselá, Havelka Jiri, and Eva Hajicová. 2004. Condition of projectivity in the underlying dependency structures. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 289–295.

- Anssi Yli-Jyrä. 2003. Multiplanarity – a model for dependency structures in treebanks. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 189–200.