Master in Artificial Intelligence

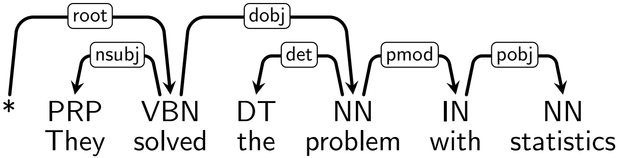
Advanced Human Language Technologies

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Outline

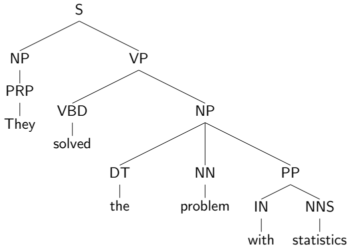
Outline

Dependency Trees



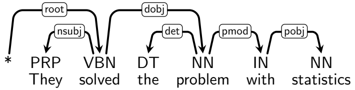
Theories of Syntactic Structure

Constituent Trees



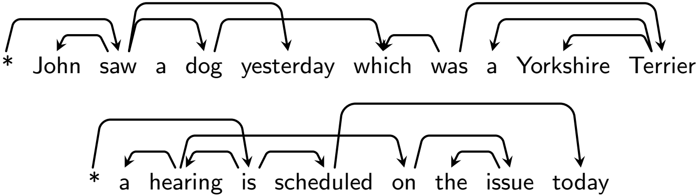
- Main element: constituents (or phrases, or bracketings)
- Constituents = abstract linguistic units
- Focus on word order
- Builds nested trees

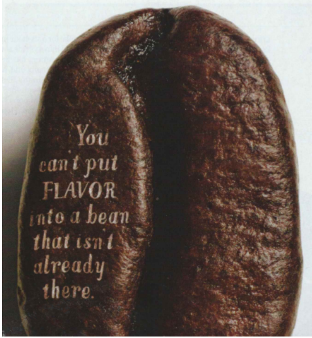
Dependency Trees



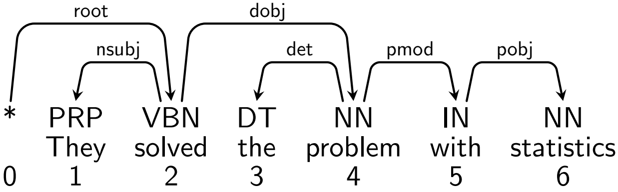
- Main element: dependency
- Focus on relations between words
- Nicely handles free word order (*fish the cat eats**) and non-projectivity (*John saw the dog yesterday which was a Yorkshire Terrier*)
- Builds dependency graphs

Non-projective dependency trees



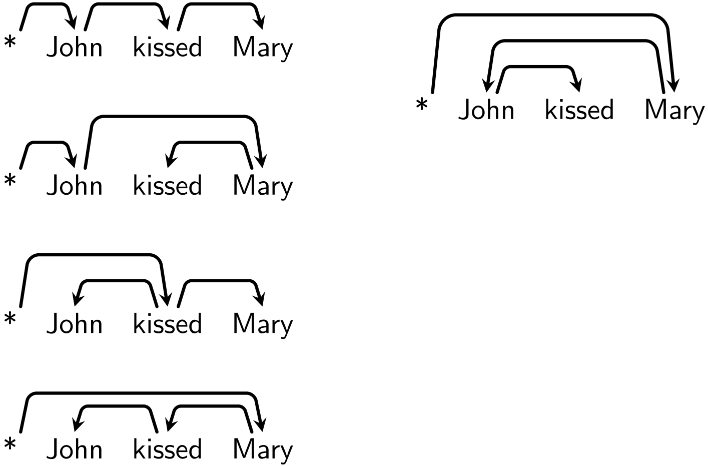


Dependency trees

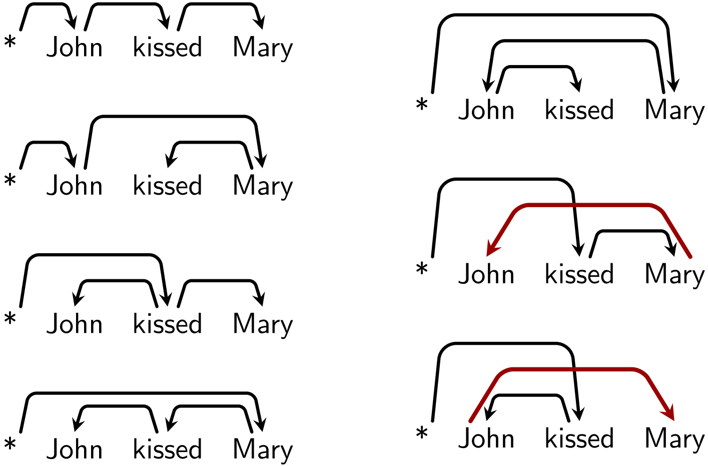


- $*$ is a special *root* symbol
- Each dependency is a tuple (h, m, l) where
  - h is the index of the head word (root is 0)
  - m is the index of the modifier word
  - l is a dependency label
  - e.g.: $(0, 2, \text{root})$, $(2, 1, \text{nsubj})$, $(2, 5, \text{dobj})$, $(4, 3, \text{det})$, $(4, 5, \text{pmod})$, $(5, 6, \text{pobj})$
- Sometimes we just consider unlabeled dependencies

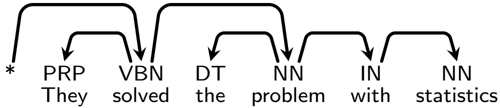
Dependency trees for “John kissed Mary”



Dependency trees for “John kissed Mary”



Conditions on Dependency Structures



- $\mathbf{y}$ is a dependency tree if:
  - (a) Each non-root token has exactly an incoming arc (i.e. one parent)
  - (b) The graph is connected
  - (c) There are no cycles
    - That is, dependency arcs form a directed tree rooted at $*$
- $\mathbf{y}$ is a projective dependency tree if:
  - Is a dependency tree
  - There are no crossing dependencies
- Note that a projective tree is also in the non-projective set –must be read as non-*necessarily*-projective

Some Notation

*	They	solved	the	problem
0	1	2	3	4

Given a sentence with n words:

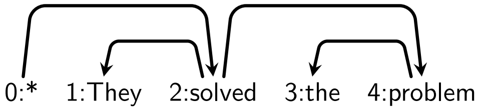
- $\mathcal{D}$ is the set of all possible dependencies that can be assigned to the sentence. Eg.

$$\begin{aligned}\mathcal{D} = \quad \{ \quad & (0,1),(0,2),(0,3),(0,4),(1,2),(1,3),(1,4) \\ & (2,1),(2,3),(2,4),(3,1),(3,2),(3,4) \\ & (4,1),(4,2),(4,3) \quad \}\end{aligned}$$

- $\mathbf{y}$ is a valid parse for s if:
  - $\mathbf{y} \subseteq \mathcal{D}$
  - $\mathbf{y}$ is a dependency tree
- $\mathcal{Y} \subseteq 2^{\mathcal{D}}$ is the set of all valid dependency trees for the sentence

Outline

Probabilistic Arc-Factored Dependency Parsing



- Assume we have $p(\text{modifier word} \mid \text{head word})$
- In a probabilistic arc-factored model:

$$
\begin{aligned}
p(\mathbf{x}, \mathbf{y}) &= p(\mathbf{x}, (*, 2), (2, 1), (2, 4), (4, 3)) \\
&= p(\mathbf{x}_2, (*, 2)\ ) \times p(\mathbf{x}, (2, 1), (2, 4), (4, 3) \mid \mathbf{x}_2, (*, 2)) \\
&= p(*) \times p(\mathbf{x}_2 \mid *) \times p(\mathbf{x}, (2, 1), (2, 4), (4, 3) \mid \mathbf{x}_2, (*, 2)) \\
&= \dots \\
&= p(\mathbf{x}_2 \mid *) \times p(\mathbf{x}_1 \mid \mathbf{x}_2) \times p(\mathbf{x}_4 \mid \mathbf{x}_2) \times p(\mathbf{x}_3 \mid \mathbf{x}_4) \\
&= \prod_{(h,m) \in \mathbf{y}} p(\mathbf{x}_m \mid \mathbf{x}_h)
\end{aligned}
$$

- Note that we assume independence between arcs

Towards Linear Arc-Factored Dependency Parsing

- Consider an arc-factored probabilistic model

$$p(\mathbf{x}, \mathbf{y}) = \prod_{(h,m)\in\mathbf{y}} p(\mathbf{x}_m \mid \mathbf{x}_h)$$

- Prediction is:

$$\begin{aligned}
\operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} p(\mathbf{x}, \mathbf{y}) &= \operatorname*{argmax}_{\mathbf{y}} \prod_{(h,m)\in\mathbf{y}} p(\mathbf{x}_m \mid \mathbf{x}_h) \\
&= \operatorname*{argmax}_{\mathbf{y}} \exp\left\{ \sum_{(h,m)\in\mathbf{y}} \log p(\mathbf{x}_m \mid \mathbf{x}_h) \right\} \\
&= \operatorname*{argmax}_{\mathbf{y}} \sum_{(h,m)\in\mathbf{y}} \log p(\mathbf{x}_m \mid \mathbf{x}_h) \\
&= \operatorname*{argmax}_{\mathbf{y}} \sum_{(h,m)\in\mathbf{y}} \mathrm{score}(\mathbf{x}, h, m)
\end{aligned}$$

where $\mathrm{score}(\mathbf{x}, h, m) = \log p(\mathbf{x}_m \mid \mathbf{x}_h)$

A CRF for Arc-Factored Dependency Parsing

- A log-linear distribution of trees $\mathbf{y}$ given $\mathbf{x}$

$$p(\mathbf{y} \mid \mathbf{x}; \mathbf{w}) = \frac{\exp(\sum_{(h,m,l)\in\mathbf{y}} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m, l))}{Z(\mathbf{x}; \mathbf{w})}$$

- $\mathbf{f}(\mathbf{x}, h, m)$ is a vector of d features of (h, m, l) assigned to x
- $\mathbf{w} \in \mathbb{R}^d$ are the parameters of the model
- $Z(\mathbf{x}; \mathbf{w}) = \sum_{\mathbf{y}\in\mathcal{Y}} \exp(\sum_{(h,m,l)\in\mathbf{y}} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m, l))$
- Prediction is linear:

$$\begin{aligned}
\operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}^*} P(\mathbf{y}|\mathbf{x}; \mathbf{w}) &= \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}^*} \frac{\exp(\sum_{(h,m,l)\in\mathbf{y}} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m, l))}{Z(\mathbf{x}; \mathbf{w})} \\
&= \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}^*} \sum_{(h,m,l)\in\mathbf{y}} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m, l)
\end{aligned}$$

Features in Arc-Factored Dependency Parsing

$\mathbf{f}(\mathbf{x}, l, h, m)$: a vector of features of (h, m, l) assigned to x

- As in PoS tagging or NERC, we typically use indicator features
- Templates in (McDonald et al 2005):

word features
h-word, h-pos
h-word
h-pos
m-word, m-pos
m-word
m-pos

dependency features
h-word, h-pos, m-word, m-pos
h-pos, m-word, m-pos
h-word, m-word, m-pos
h-word, h-pos, m-pos
h-word, h-pos, m-word
h-word, m-word
h-pos, m-pos

- Example: (feature template + dependency direction)

$$\mathbf{f}_j(\mathbf{x}, h, m, l) = \begin{cases} 1 & \text{if } \mathrm{word}(h) = \textit{solve} \text{ and } \mathrm{word}(m) = \textit{problem} \\ & \quad \text{and } l = \textit{dobj} \text{ and } h < m \\ 0 & \text{otherwise} \end{cases}$$

A CRF for Arc-Factored Dependency Parsing

$$p(\mathbf{y} \mid \mathbf{x}; \mathbf{w}) = \frac{\exp(\sum_{(h,m,l)\in\mathbf{y}} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m, l))}{Z(\mathbf{x}; \mathbf{w})}$$

- Parameter estimation: Learn parameters $\mathbf{w}$ given training data

$$\left\{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots, (\mathbf{x}^{(m)}, \mathbf{y}^{(m)})\right\}$$

- Decoding: predict the best dependency tree for $\mathbf{x}$

$$\operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} \mathrm{P}(\mathbf{y}|\mathbf{x}; \mathbf{w})$$

when

- $\mathcal{Y}$ is the set of projective trees for $\mathbf{x}$
- $\mathcal{Y}$ is the set of non-projective trees for $\mathbf{x}$

Parameter Estimation: CRFs for Parsing

...analogous to CRFs for Tagging

- Goal: Estimate $\mathbf{w}$ given a training set

$$\left\{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots, (\mathbf{x}^{(m)}, \mathbf{y}^{(m)})\right\}$$

- Define the conditional log-likelihood of the data:

$$L(\mathbf{w}) = \frac{1}{m} \sum_{k=1}^{m} \log \mathrm{P}(\mathbf{y}^{(k)} | \mathbf{x}^{(k)}; \mathbf{w})$$

  $L(\mathbf{w})$ measures how well $\mathbf{w}$ explains the data. A good value for $\mathbf{w}$ will give a high value for $\mathrm{P}(\mathbf{y}^{(k)} | \mathbf{x}^{(k)}; \mathbf{w})$ for all training examples $k = 1 \ldots m$.

- We want $\mathbf{w}$ that maximizes $L(\mathbf{w})$

Learning the Parameters of a CRF

. . . analogous to CRFs for Tagging

- Consider a regularized objective:

$$\mathbf{w}^* = \operatorname*{argmax}_{\mathbf{w} \in \mathbb{R}^D} L(\mathbf{w}) - \frac{\lambda}{2} ||\mathbf{w}||^2$$

  where
  - The first term is the log-likelihood of the data
  - The second term is a regularization term, it penalizes solutions with large norm
  - λ is a parameter to control the trade-off between fitting the data and model complexity

Learning the Parameters of a CRF

. . . analogous to CRFs for Tagging

- Find
$$\mathbf{w}^* = \operatorname*{argmax}_{\mathbf{w} \in \mathbb{R}^D} L(\mathbf{w}) - \frac{\lambda}{2}||\mathbf{w}||^2$$
- In general there is no analytical solution to this optimization
- We use iterative techniques, i.e. gradient-based optimization
  1. Initialize $\mathbf{w} = \mathbf{0}$
  2. Take derivatives of $L(\mathbf{w}) - \frac{\lambda}{2}||\mathbf{w}||^2$, compute gradient
  3. Move $\mathbf{w}$ in steps proportional to the gradient
  4. Repeat steps 2 and 3 until convergence

Computing the gradient

... analogous to CRFs for Tagging

$$\frac{\partial L(\mathbf{w})}{\partial \mathbf{w}_j} = \frac{1}{m}\sum_{k=1}^{m} \mathbf{f}_j(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) - \sum_{k=1}^{m} \sum_{\mathbf{y} \in \mathcal{Y}^*} \mathrm{P}(\mathbf{y}|\mathbf{x}^{(k)}; \mathbf{w})\ \mathbf{f}_j(\mathbf{x}^{(k)}, \mathbf{y})$$

where

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = \sum_{(h,m,l) \in \mathbf{y}} \mathbf{f}_j(\mathbf{x}, h, m, l)$$

- First term: observed mean feature value
- Second term: expected feature value under current $\mathbf{w}$

Computing the gradient

. . . analogous to CRFs for Tagging

- The first term is easy to compute, by counting explicitly

$$\frac{1}{m}\sum_{k=1}^{m}\sum_{(h,m,l)\in\mathbf{y}^{(k)}}\mathbf{f}_j(\mathbf{x},h,m,l)$$

- The second term is more involved,

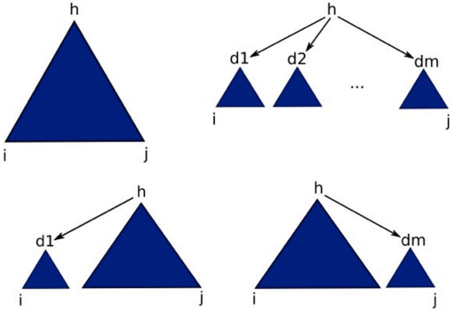
$$\sum_{k=1}^{m}\sum_{\mathbf{y}\in\mathcal{Y}}\mathrm{P}(\mathbf{y}|\mathbf{x}^{(k)};\mathbf{w})\sum_{(h,m,l)\in\mathbf{y}}\mathbf{f}_j(\mathbf{x}^{(k)},h,m,l)$$

  because it sums over all trees $\mathbf{y}\in\mathcal{Y}$
- There exist efficient algorithms for summing over $\mathcal{Y}$, both for projective and non-projective sets of trees

Outline

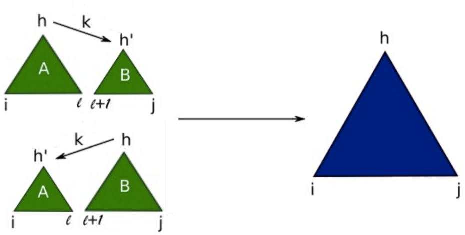
Parsing Projective Structures (I)

- Any projective tree can be written as the combination of:
  - two smaller *adjacent* projective trees and
  - a dependency connecting their roots



Parsing Projective Structures (II)

- The algorithm is a variation of CKY
- $\pi[i,j,h]$: score of dependency tree from i to j with head h

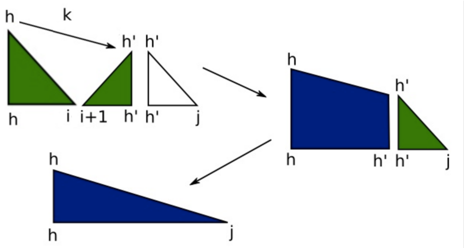


$$\pi[i,j,h] = \max_{\substack{i \le l < j \\ 1 \le k \le K}} \left\{ \begin{array}{l} \max\limits_{l < h' \le j} \pi[i,l,h] + \pi[l+1,j,h'] + \mathbf{w} \cdot \mathbf{f}(\mathbf{x},h,h') \quad , \\ \max\limits_{i \le h' \le l} \pi[i,l,h'] + \pi[l+1,j,h] + \mathbf{w} \cdot \mathbf{f}(\mathbf{x},h,h') \end{array} \right\}$$

- Cost: $O(Kn^5)$

Parsing Projective Structures (III)

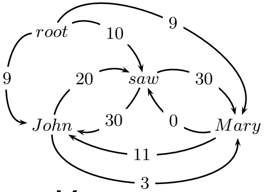
- (Eisner 1996), (Eisner 2000): an algorithm in $O(Kn^3)$
- Main idea: split constituents in half so that heads are at the boundary



Outline

Parsing Non-Projective Structures

- (McDonald et al 2005): non-projective parsing as maximum-spanning trees, using the Chu-Liu-Edmonds algorithm

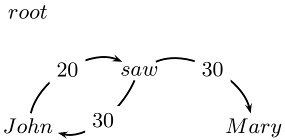


- Example for *John saw Mary*
- Build a graph:
  - Nodes are tokens (and the root token)
  - A weighted directed edge between any two vertices

$$w_{i,j} = \max_{1 \leq k \leq K} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, i, j, k)$$

Chu-Liu-Edmonds, example

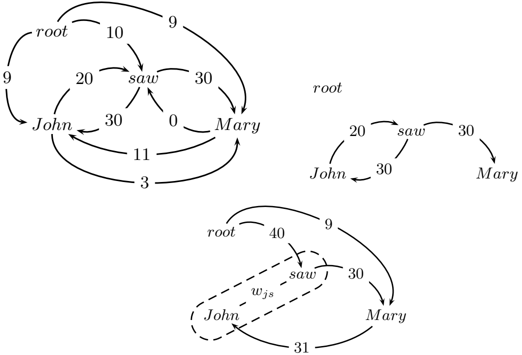
- Step 1: for each word, find highest-scoring incoming edge



- If we get a tree, we have found the MST
- If not, there has to be a cycle

Chu-Liu-Edmonds, example

- Step 2: identify cycle and *contract* it into a new node c



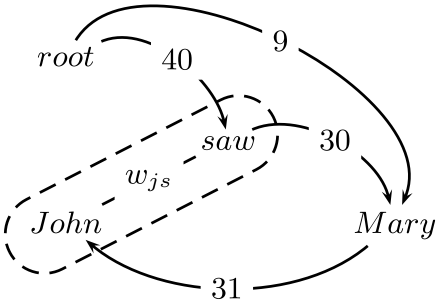
- Weight of edges between c and other nodes i:
  - $c \rightarrow i$: max weight of any node in c to i
  - $i \rightarrow c$: max weight of tree with root i that spans c

    root $\rightarrow$ *saw* $\rightarrow$ *John* : 40

    root $\rightarrow$ *John* $\rightarrow$ *saw* : 29

Chu-Liu-Edmonds

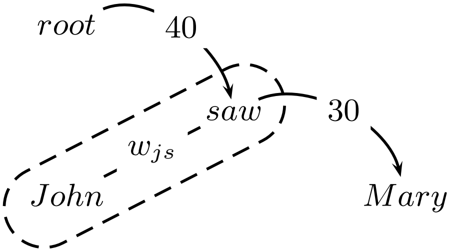
- Theorem (Leonidas 2003): the weight of the MST on the contracted graph is equal to the weight of the MST in the original graph



- Recursively call the algorithm on the new graph

Chu-Liu-Edmonds

- After one recursive call we get



- It is a tree! (if not, contract and recurse)
- The original MST can be reconstructued by undoing the contraction operations (see (McDonald et al 2005) for details)
- Cost: $O(n^3)$ (naive), $O(n^2)$ (improved)

Outline

Transition-Based parsers

- Inspired on shift-reduce parsers.
- The parser has a current state or configuration consisting of a stack (of tokens processed and tree built so far) and a buffer (tokens remaining).
- At each step, a transition is chosen to alter the configuration and move.
- Parsing stops when a final configuration is reached
- No backtracking, cost is $\mathcal{O}(n)$

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP$\rightarrow$DT NN

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP$\rightarrow$DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP$\rightarrow$DT NN
NP Vt NP	IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP$\rightarrow$DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP$\rightarrow$DT NN
NP Vt NP	IN DT NN	*reduce VP$\rightarrow$Vt NP
NP VP	IN DT NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		reduce PP→IN NP

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		reduce PP→IN NP
NP VP PP		

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		reduce PP→IN NP
NP VP PP		reduce VP→VP PP

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		reduce PP→IN NP
NP VP PP		reduce VP→VP PP
NP VP		

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		reduce PP→IN NP
NP VP PP		reduce VP→VP PP
NP VP		reduce S→NP VP

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		reduce PP→IN NP
NP VP PP		reduce VP→VP PP
NP VP		reduce S→NP VP
S		

Shift-Reduce Parsing Example

The	woman	saw	the	man	with	the	telescope
DT	NN	Vt	DT	NN	IN	DT	NN

Stack	Buffer	Transition
	DT NN Vt DT NN IN DT NN	shift
DT	NN Vt DT NN IN DT NN	shift
DT NN	Vt DT NN IN DT NN	reduce NP→DT NN
NP	Vt DT NN IN DT NN	shift
NP Vt	DT NN IN DT NN	shift
NP Vt DT	NN IN DT NN	shift
NP Vt DT NN	IN DT NN	reduce NP→DT NN
NP Vt NP	IN DT NN	*reduce VP→Vt NP
NP VP	IN DT NN	shift
NP VP IN	DT NN	shift
NP VP IN DT	NN	shift
NP VP IN DT NN		reduce NP→DT NN
NP VP IN NP		reduce PP→IN NP
NP VP PP		reduce VP→VP PP
NP VP		reduce S→NP VP
S		stop

Transition-Based parsers

- Only one tree is produced: Not suitable for ambiguous grammars (common in NLP)
- We can add probabilities to select which transition is selected at each step: Similar to CKY with PCFGs, but greedy search (may be made less greedy with e.g. beam-search)
- Or better: we can add features and use ML to take the decision.

Let's see how it is applied to dependency parsing

Arc-Standard algorithm

- A configuration (S, B, A) of the parser consists of:
  - A stack S containing seen words
  - A buffer B containing not-yet seen words
  - The dependency graph A built so far (not a tree yet)
- Initial configuration: $([\,], [0 \ldots n], [\,])$
- Final configuration: $([0], [\,], A)$
- Possible transitions:
  - shift: push next word in the buffer onto the stack
  - left-arc: add an arc from $S[0]$ to $S[1]$ and remove $S[1]$ from the stack
  - right-arc: add an arc from $S[1]$ to $S[0]$ and remove $S[0]$ from the stack

Arc-Standard Transition definitions

- shift (sh)
  $(\sigma, [i|\beta], A) \Rightarrow ([\sigma|i], \beta, A)$
- left-arc (la-L)
  $([\sigma|i|j], B, A) \Rightarrow ([\sigma|j], B, A \cup \{j, i, L\})$
- right-arc (ra-L): $([\sigma|i|j], B, A) \Rightarrow ([\sigma|i], B, A \cup \{i, j, L\})$

Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	

* the woman saw the man with glasses

Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh

* the woman saw the man with glasses

Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	

* the woman saw the man with glasses

Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh

* the woman saw the man with glasses

Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	

* the woman saw the man with glasses

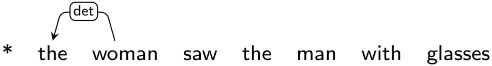
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det

* the woman saw the man with glasses

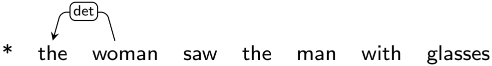
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	



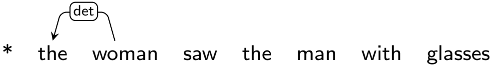
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh



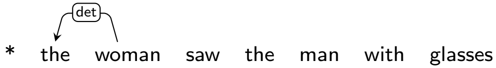
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	



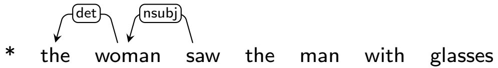
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj



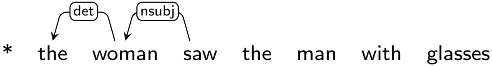
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	



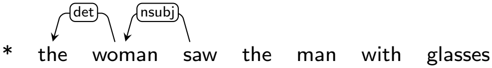
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh



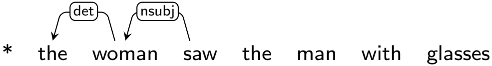
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	



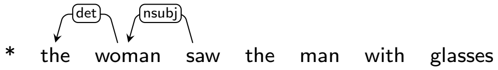
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh



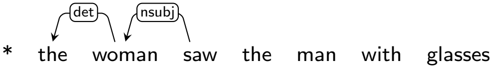
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	



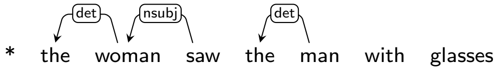
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det



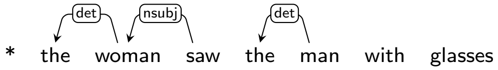
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	



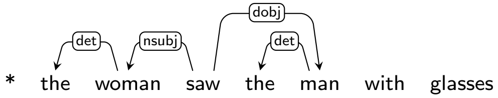
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj



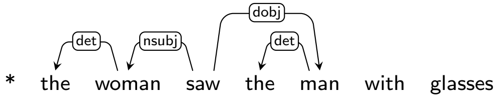
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	



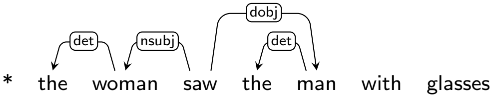
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh



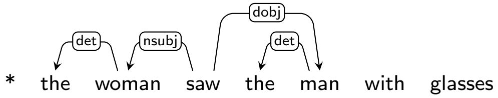
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	



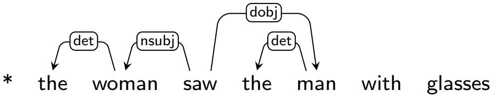
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh



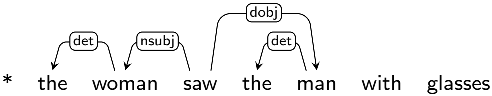
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		



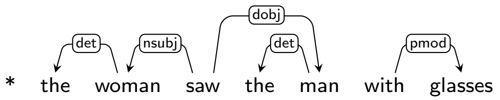
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		ra-pmod



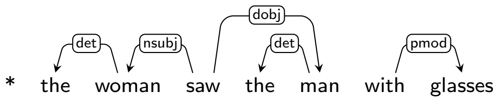
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		ra-pmod
* saw with		



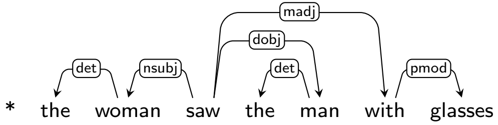
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		ra-pmod
* saw with		ra-madj



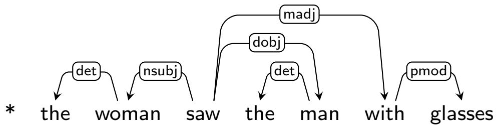
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		ra-pmod
* saw with		ra-madj
* saw		



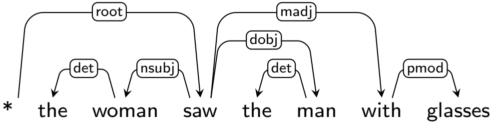
Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		ra-pmod
* saw with		ra-madj
* saw		ra-root



Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		ra-pmod
* saw with		ra-madj
* saw		ra-root
*		

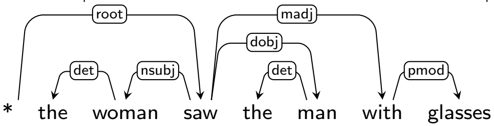


Dependency Parsing

Transition-Based parsers

Arc-Standard Example

Stack	Buffer	Transition
	* the woman saw the man with glasses	sh
* the	woman saw the man with glasses	sh
* the woman	saw the man with glasses	la-det
* woman	saw the man with glasses	sh
* woman saw	the man with glasses	la-subj
* saw	the man with glasses	sh
* saw the	man with glasses	sh
* saw the man	with glasses	la-det
* saw man	with glasses	ra-dobj
* saw	with glasses	sh
* saw with	glasses	sh
* saw with glasses		ra-pmod
* saw with		ra-madj
* saw		ra-root
*		stop



Alternative Transition Models

- Stack-stack arcs
  - Arc-standard (shift, left-arc, right-arc)
  - Non-projective (shift, swap, left-arc, right-arc)
- Stack-buffer arcs
  - Arc-eager (shift, reduce, left-arc, right-arc)
  - Arc-standard variant (shift, left-arc, right-arc)

Transition Selection

- Classifier that produces the best transition for the current configuration
- Too many possible configurations: Need to model them as feature vectors and use ML:
- Typical features:
  - word/lemma/PoS for $S[0]$, $S[1]$, $B[0]$, $B[1]$
  - morphological features (gender, number, mode, tense, etc) in $S[0]$, $B[0]$
  - number of children of $S[0]$
  - dependency labels of $S[0]$ children
  - ..etc
- We can use SVM, perceptron, MBL, DT, ... any feature-based ML classifier

Transition Selection

- Classifier that produces the best transition for the current configuration
- Too many possible configurations: Need to model them as feature vectors and use ML:
- Typical features:
  - word/lemma/PoS for $S[0]$, $S[1]$, $B[0]$, $B[1]$
  - morphological features (gender, number, mode, tense, etc) in $S[0]$, $B[0]$
  - number of children of $S[0]$
  - dependency labels of $S[0]$ children
  - ..etc
- We can use SVM, perceptron, MBL, DT, ... any feature-based ML classifier

... or we can use Deep Learning