Dependency Parsing
Lecture 9

Jakub Waszczuk

Heinrich Heine Universität

Summer semester 2018

Today

Problems:

- Spurious ambiguity (several optimal transition sequences possible)
- Error propagation (single mistake causes more mistakes)
- Parser not trained with suboptimal decisions

Solutions:

- Beam search: consider multiple alternatives
- Easy first: give up left-to-right incremental search
- Dynamic oracles: learn from errors at train time

(with slides from EACL 2014 tutorial by McDonald & Nivre)

Today

Problems:

- Spurious ambiguity (several optimal transition sequences possible)
- Error propagation (single mistake causes more mistakes)
- Parser not trained with suboptimal decisions

Solutions:

- Beam search: consider multiple alternatives
- Easy first: give up left-to-right incremental search
- **Dynamic oracles: learn from errors at train time**

(with slides from EACL 2014 tutorial by McDonald & Nivre)

Online Learning with a Conventional Oracle

```
Learn($\{T_1, \ldots, T_N\}$)
1   $\mathbf{w} \leftarrow 0.0$
2   for $i$ in $1..K$
3       for $j$ in $1..N$
4           $c \leftarrow ([\ ], [0, 1, \ldots, n_j], \{\ \})$
5           while $B_c \neq [\ ]$
6               $t^* \leftarrow \text{argmax}_t\, \mathbf{w} \cdot \mathbf{f}(c, t)$
7               $t_o \leftarrow o(c, T_i)$
8               if $t^* \neq t_o$
9                   $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{f}(c, t_o) - \mathbf{f}(c, t^*)$
10              $c \leftarrow t_o(c)$
11  return $\mathbf{w}$
```

Online Learning with a Conventional Oracle

```
Learn($\{T_1, \ldots, T_N\}$)
1   $\mathbf{w} \leftarrow 0.0$
2   for $i$ in $1..K$
3       for $j$ in $1..N$
4           $c \leftarrow ([\,], [0, 1, \ldots, n_j], \{\,\})$
5           while $B_c \neq [\,]$
6               $t^* \leftarrow \text{argmax}_t\, \mathbf{w} \cdot \mathbf{f}(c, t)$
7               $t_o \leftarrow o(c, T_i)$
8               if $t^* \neq t_o$
9                   $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{f}(c, t_o) - \mathbf{f}(c, t^*)$
10              $c \leftarrow t_o(c)$
11  return $\mathbf{w}$
```

T_j

- Oracle $o(c, T_i)$ returns the optimal transition for c and T_i

Conventional Oracle for Arc-Eager Parsing

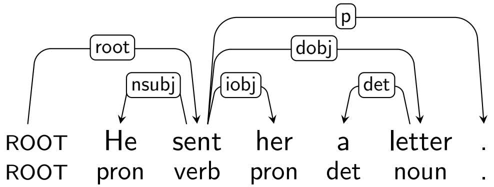
$$o(c, T) = \begin{cases} \text{Left-Arc} & \text{if top}(S_c) \leftarrow \text{first}(B_c) \text{ in } T \\ \text{Right-Arc} & \text{if top}(S_c) \rightarrow \text{first}(B_c) \text{ in } T \\ \text{Reduce} & \text{if } \exists v < \text{top}(S_c) : v \leftrightarrow \text{first}(B_c) \text{ in } T \\ \text{Shift} & \text{otherwise} \end{cases}$$

- Correct:
  - Derives T in a configuration sequence $C_{o,T} = c_0, \ldots, c_m$
- Problems:
  - Deterministic: Ignores other derivations of T
  - Incomplete: Valid only for configurations in $C_{o,T}$

Oracle Parse

Transitions:

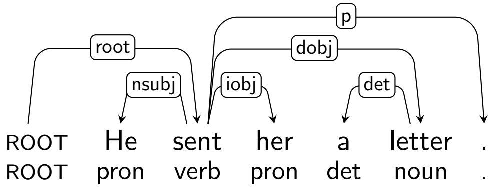
Stack	**Buffer**	**Arcs**
[ ]	[ROOT, He, sent, her, a, letter, .]	



Oracle Parse

Transitions: SH

Stack	**Buffer**	**Arcs**
[ROOT]	[He, sent, her, a, letter, .]	



Oracle Parse

Transitions: SH-RA

Stack

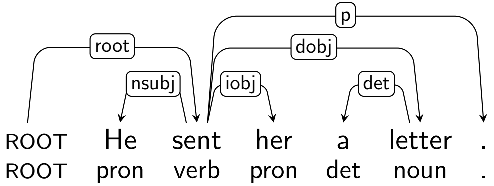
[ROOT, He]

Buffer

[sent, her, a, letter, .]

Arcs

$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$



Oracle Parse

Transitions: SH-RA-LA

Stack

[ROOT]

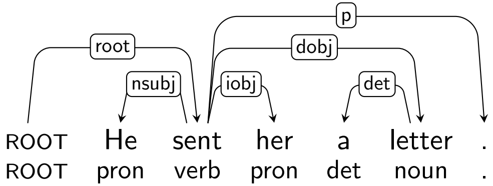
Buffer

[sent, her, a, letter, .]

Arcs

ROOT $\xrightarrow{\text{root}}$ sent

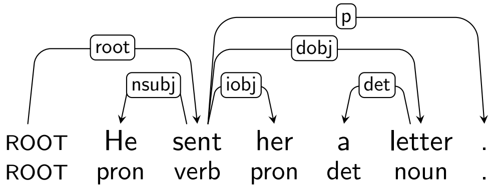
He $\xleftarrow{\text{sbj}}$ sent



Oracle Parse

Transitions: SH-RA-LA-SH

Stack	**Buffer**	**Arcs**
[ROOT, sent]	[her, a, letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$



Oracle Parse

Transitions: SH-RA-LA-SH-RA

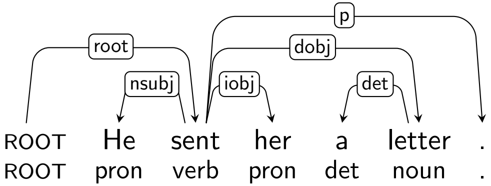
Stack	**Buffer**
[ROOT, sent, her]	[a, letter, .]

Arcs

$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$



Oracle Parse

Transitions: SH-RA-LA-SH-RA-SH

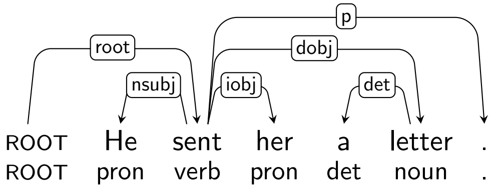
Stack	**Buffer**
[ROOT, sent, her, a]	[letter, .]

Arcs

$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

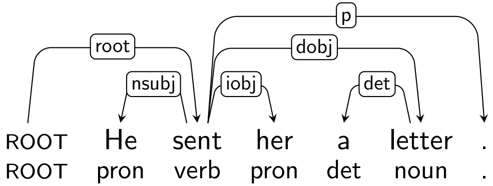
$\text{sent} \xrightarrow{\text{iobj}} \text{her}$



Oracle Parse

Transitions: SH-RA-LA-SH-RA-SH-LA

Stack	**Buffer**	**Arcs**
[ROOT, sent, her]	[letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{sent} \xrightarrow{\text{iobj}} \text{her}$
		$\text{a} \xleftarrow{\text{det}} \text{letter}$



Oracle Parse

Transitions: SH-RA-LA-SH-RA-SH-LA-RE

Stack

[ROOT, sent]

Buffer

[letter, .]

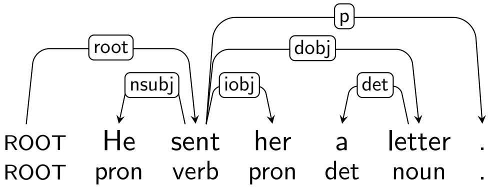
Arcs

$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$



Oracle Parse

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA

Stack	Buffer
[ROOT, sent, letter]	[.]

Arcs

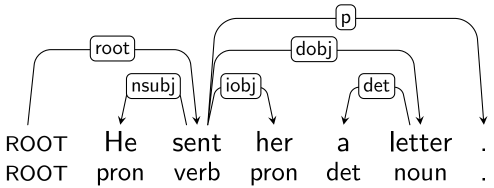
$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$

$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$



Oracle Parse

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE

Stack [ROOT, sent]

Buffer [.]

Arcs

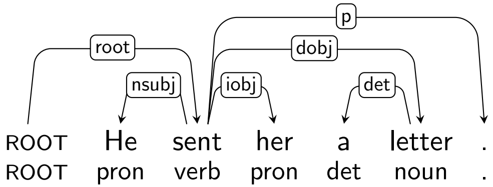
$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$

$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$



Oracle Parse

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

Stack	**Buffer**
[ROOT, sent, .]	[ ]

Arcs

$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

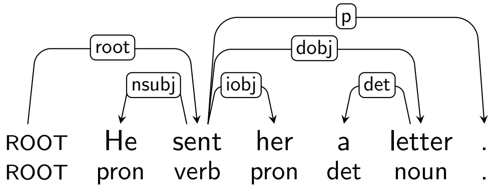
$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$

$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$

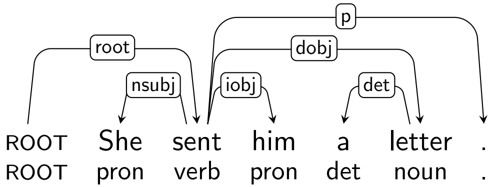
$\text{sent} \xrightarrow{\text{p}} .$



Non-Determinisim

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA
SH-RA-LA-SH-RA

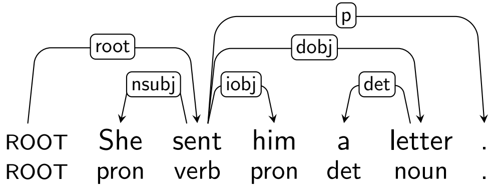
Stack	**Buffer**	**Arcs**
[ROOT, sent, her]	[a, letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{sent} \xrightarrow{\text{iobj}} \text{her}$



Non-Determinisim

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA
SH-RA-LA-SH-RA-RE

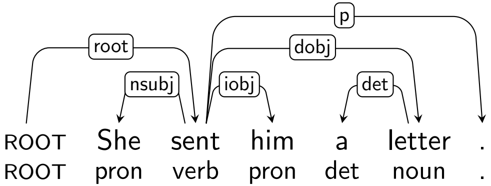
Stack	**Buffer**	**Arcs**
[ROOT, sent]	[a, letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{sent} \xrightarrow{\text{iobj}} \text{her}$



Non-Determinisim

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA
SH-RA-LA-SH-RA-RE-SH

Stack	**Buffer**	**Arcs**
[ROOT, sent, a]	[letter, .]	ROOT $\xrightarrow{\text{root}}$ sent
		He $\xleftarrow{\text{sbj}}$ sent
		sent $\xrightarrow{\text{iobj}}$ her

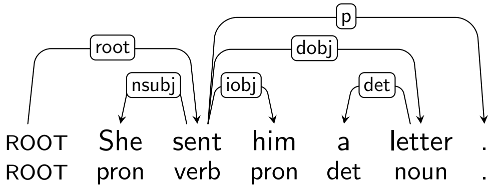


Non-Determinisim

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

SH-RA-LA-SH-RA-RE-SH-LA

Stack	**Buffer**	**Arcs**
[ROOT, sent]	[letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{sent} \xrightarrow{\text{iobj}} \text{her}$
		$\text{a} \xleftarrow{\text{det}} \text{letter}$



Non-Determinisim

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA
SH-RA-LA-SH-RA-RE-SH-LA-RA

Stack	**Buffer**
[ROOT, sent, letter]	[.]

Arcs

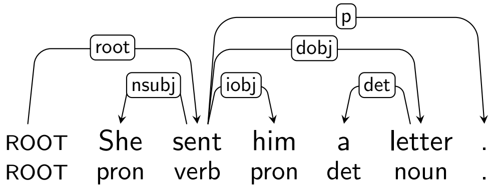
$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$

$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$



Non-Determinisim

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA
SH-RA-LA-SH-RA-RE-SH-LA-RA-RE

Stack	**Buffer**
[ROOT, sent]	[.]

Arcs

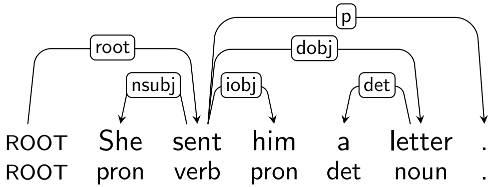
$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$

$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$



Non-Determinisim

Transitions: SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA
SH-RA-LA-SH-RA-RE-SH-LA-RA-RE-RA

Stack	**Buffer**
[ROOT, sent, .]	[ ]

Arcs

$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

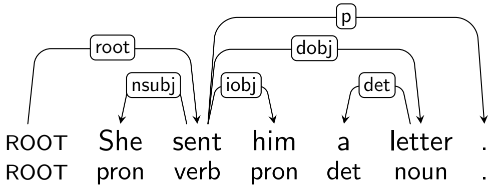
$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{sent} \xrightarrow{\text{iobj}} \text{her}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$

$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$

$\text{sent} \xrightarrow{\text{p}} .$

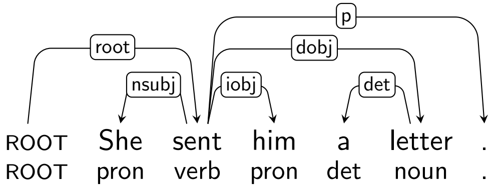


Non-Optimality

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

Transitions: SH-RA-LA-SH

Stack	**Buffer**	**Arcs**
[ROOT, sent]	[her, a, letter, .]	ROOT $\xrightarrow{\text{root}}$ sent
		He $\xleftarrow{\text{sbj}}$ sent

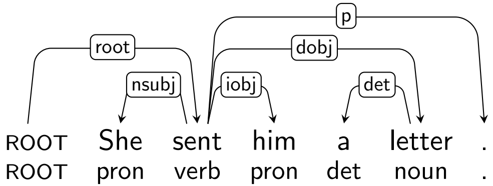


Non-Optimality

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

Transitions: SH-RA-LA-SH-SH

Stack	**Buffer**	**Arcs**
[ROOT, sent, her]	[a, letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

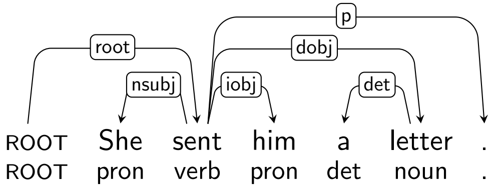


Non-Optimality

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

Transitions: SH-RA-LA-SH-SH-SH

Stack	**Buffer**	**Arcs**
[ROOT, sent, her, a]	[letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$



Non-Optimality

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

Transitions: SH-RA-LA-SH-SH-SH-LA

Stack

[ROOT, sent, her]

Buffer

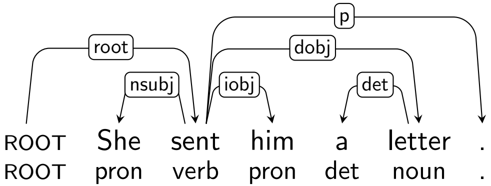
[letter, .]

Arcs

$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$

$\text{He} \xleftarrow{\text{sbj}} \text{sent}$

$\text{a} \xleftarrow{\text{det}} \text{letter}$

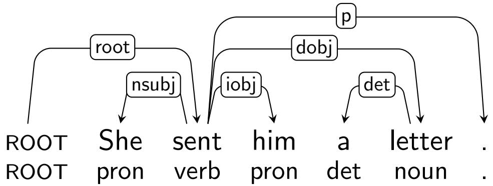


Non-Optimality

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

Transitions: SH-RA-LA-SH-SH-SH-LA-SH

Stack	**Buffer**	**Arcs**
[ROOT, sent, her, letter]	[.]	ROOT $\xrightarrow{\text{root}}$ sent
		He $\xleftarrow{\text{sbj}}$ sent
		a $\xleftarrow{\text{det}}$ letter

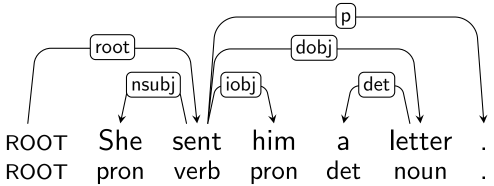


Non-Optimality

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

Transitions: SH-RA-LA-SH-SH-SH-LA-SH-SH [3/6]

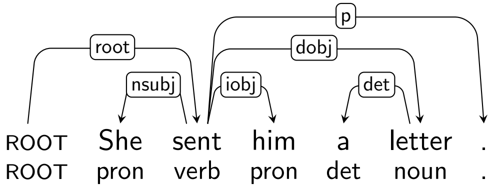
Stack	**Buffer**	**Arcs**
[ROOT, sent, letter, .]	[ ]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{a} \xleftarrow{\text{det}} \text{letter}$



Non-Optimality

	SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA	
Transitions:	SH-RA-LA-SH-SH-SH-LA-SH-SH	[3/6]
	SH-RA-LA-SH-SH-SH-LA	

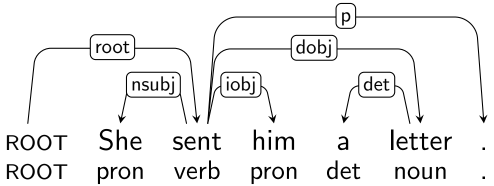
Stack	**Buffer**	**Arcs**
[ROOT, sent, her]	[letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{a} \xleftarrow{\text{det}} \text{letter}$



Non-Optimality

	SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA	
Transitions:	SH-RA-LA-SH-SH-SH-LA-SH-SH	[3/6]
	SH-RA-LA-SH-SH-SH-LA-LA	

Stack	**Buffer**	**Arcs**
[ROOT, sent]	[letter, .]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{a} \xleftarrow{\text{det}} \text{letter}$
		$\text{her} \xleftarrow{?} \text{letter}$



Non-Optimality

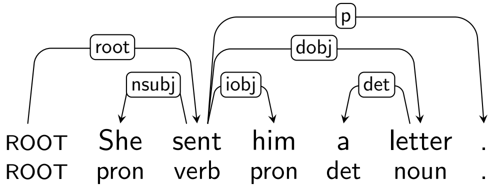
Transitions:

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

SH-RA-LA-SH-SH-SH-LA-SH-SH [3/6]

SH-RA-LA-SH-SH-SH-LA-LA-RA

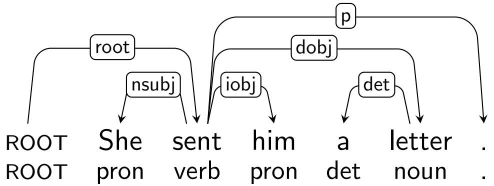
Stack	**Buffer**	**Arcs**
[ROOT, sent, letter]	[.]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{a} \xleftarrow{\text{det}} \text{letter}$
		$\text{her} \xleftarrow{?} \text{letter}$
		$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$



Non-Optimality

	SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA	
Transitions:	SH-RA-LA-SH-SH-SH-LA-SH-SH	[3/6]
	SH-RA-LA-SH-SH-SH-LA-LA-RA-RE	

Stack	**Buffer**	**Arcs**
[ROOT, sent]	[.]	ROOT $\xrightarrow{\text{root}}$ sent
		He $\xleftarrow{\text{sbj}}$ sent
		a $\xleftarrow{\text{det}}$ letter
		her $\xleftarrow{?}$ letter
		sent $\xrightarrow{\text{dobj}}$ letter



Non-Optimality

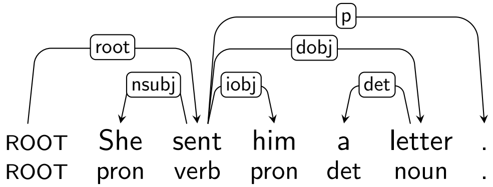
Transitions:

SH-RA-LA-SH-RA-SH-LA-RE-RA-RE-RA

SH-RA-LA-SH-SH-SH-LA-SH-SH [3/6]

SH-RA-LA-SH-SH-SH-LA-LA-RA-RE-RA [5/6]

Stack	**Buffer**	**Arcs**
[ROOT, sent, .]	[ ]	$\text{ROOT} \xrightarrow{\text{root}} \text{sent}$
		$\text{He} \xleftarrow{\text{sbj}} \text{sent}$
		$\text{a} \xleftarrow{\text{det}} \text{letter}$
		$\text{her} \xleftarrow{?} \text{letter}$
		$\text{sent} \xrightarrow{\text{dobj}} \text{letter}$
		$\text{sent} \xrightarrow{\text{p}} .$



Dynamic Oracles

- Optimality:
  - A transition is optimal if the best tree remains reachable
  - Best tree = $\text{argmin}_{T'} \mathcal{L}(T, T')$
- Oracle:
  - Boolean function $o(c, t, T) =$ **true** if t is optimal for c and T
  - Non-deterministic: More than one transition can be optimal
  - Complete: Correct for all configurations
- New problem:
  - How do we know which trees are reachable?

Reachability for Arcs and Trees

- Arc reachability:
  - An arc $w_i \rightarrow w_j$ is reachable in c iff $w_i \rightarrow w_j \in A_c$, or $w_i \in S_c \cup B_c$ and $w_j \in B_c$ (same for $w_i \leftarrow w_j$)
- Tree reachability:
  - A (projective) tree T is reachable in c iff every arc in T is reachable in c
- Arc-decomposable systems [Goldberg and Nivre 2013]:
  - Tree reachability reduces to arc reachability
  - Holds for some transition systems but not all
    - Arc-eager and easy-first are arc-decomposable
    - Arc-standard is not decomposable

Oracles for Arc-Decomposable Systems

$$o(c, t, T) = \begin{cases} \textbf{true} & \text{if } [\mathcal{R}(c) - \mathcal{R}(t(c))] \cap T = \emptyset \\ \textbf{false} & \text{otherwise} \end{cases}$$

$$\text{where } \mathcal{R}(c) \equiv \{a \mid a \text{ is an arc reachable in } c\}$$

Arc-Eager

$$o(c, \text{LA}, T) = \begin{cases} \textbf{false} & \text{if } \exists w \in B_c : s \leftrightarrow w \in T \text{ (except } s \leftarrow b) \\ \textbf{true} & \text{otherwise} \end{cases}$$

$$o(c, \text{RA}, T) = \begin{cases} \textbf{false} & \text{if } \exists w \in S_c : w \leftrightarrow b \in T \text{ (except } s \rightarrow b) \\ \textbf{true} & \text{otherwise} \end{cases}$$

$$o(c, \text{RE}, T) = \begin{cases} \textbf{false} & \text{if } \exists w \in B_c : s \rightarrow w \in T \\ \textbf{true} & \text{otherwise} \end{cases}$$

$$o(c, \text{SH}, T) = \begin{cases} \textbf{false} & \text{if } \exists w \in S_c : w \leftrightarrow b \in T \\ \textbf{true} & \text{otherwise} \end{cases}$$

Notation: s = node on top of the stack S
b = first node in the buffer B

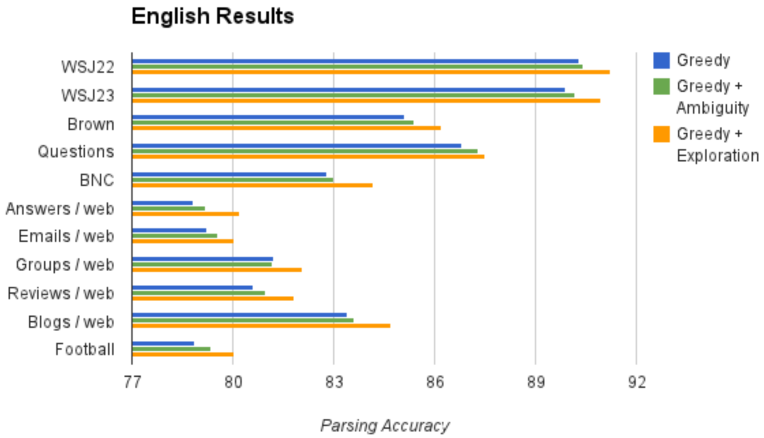
Online Learning with a Dynamic Oracle

```
Learn($\{T_1, \ldots, T_N\}$)
 1  $\mathbf{w} \leftarrow 0.0$
 2  for $i$ in $1..K$
 3      for $j$ in $1..N$
 4          $c \leftarrow ([\,]_S, [w_1, \ldots, w_{n_j}]_B, \{\,\})$
 5          while $B_c \neq [\,]$
 6              $t^* \leftarrow \text{argmax}_t\, \mathbf{w} \cdot \mathbf{f}(c, t)$
 7              $t_o \leftarrow \text{argmax}_{t \in \{t | o(c,t,T_i)\}}\, \mathbf{w} \cdot \mathbf{f}(c, t)$
 8              if $t^* \neq t_o$
 9                  $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{f}(c, t_o) - \mathbf{f}(c, t^*)$
10              $c \leftarrow \text{choice}(t_o(c), t^*(c))$
11  return $\mathbf{w}$
```

Online Learning with a Dynamic Oracle

```
Learn($\{T_1, \ldots, T_N\}$)
    $\mathbf{w} \leftarrow 0.0$
    **for** $i$ in $1..K$
        **for** $j$ in $1..N$
            $c \leftarrow ([\,]_S, [w_1, \ldots, w_{n_j}]_B, \{\,\})$
            **while** $B_c \neq [\,]$
                $t^* \leftarrow \text{argmax}_t\, \mathbf{w} \cdot \mathbf{f}(c, t)$
                $t_o \leftarrow \text{argmax}_{t \in \{t | o(c,t,T_i)\}}\, \mathbf{w} \cdot \mathbf{f}(c, t)$
                **if** $t^* \neq t_o$
                    $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{f}(c, t_o) - \mathbf{f}(c, t^*)$
                $c \leftarrow \text{choice}(t_o(c), t^*(c))$
    **return** $\mathbf{w}$
```

- Ambiguity: use model score to break ties
- Exploration: follow model prediction even if not optimal



[Goldberg and Nivre 2012]

T H E

E N D

References and Further Reading

- 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.
- Bernd Bohnet and Jonas Kuhn. 2012. The best of both worlds – a graph-based completion model for transition-based parsers. In *Proceedings of the 13th Conference of the European Chpater of the Association for Computational Linguistics (EACL)*, pages 77–87.
- Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465.
- Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajič. 2013. Joint morphological and syntactic analysis for richly inflected languages. *Transactions of the Association for Computational Linguistics*, 1:415–428.
- Jinho D. Choi and Andrew McCallum. 2013. Transition-based dependency parsing with selectional branching. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1052–1062.

- Jinho D. Choi and Martha Palmer. 2011. Getting the most out of transition-based dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 687–692.
- Shay B. Cohen, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Exact inference for generative probabilistic non-projective dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1245.
- Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 112–119.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–8.
- Michael A. Covington. 2001. A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102.
- Yoav Goldberg and Michael Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT)*, pages 742–750.

- Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *Proceedings of COLING 2012*, pages 959–976.
- Yoav Goldberg and Joakim Nivre. 2013. Training deterministic parsers with non-deterministic oracles. *Transactions of the Association for Computational Linguistics*, 1:403–414.
- Carlos Gómez-Rodríguez and Joakim Nivre. 2010. A transition-based parser for 2-planar dependency structures. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1492–1501.
- Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental joint pos tagging and dependency parsing in chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 1216–1224.
- Matthew Honnibal, Yoav Goldberg, and Mark Johnson. 2013. A non-monotonic arc-eager transition system for dependency parsing. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 163–172.
- Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1077–1086.

- Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–151.
- Richard Johansson and Pierre Nugues. 2006. Investigating multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL)*, pages 206–210.
- Marco Kuhlmann, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Dynamic programming algorithms for transition-based dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 673–682.
- John Lee, Jason Naradowsky, and David A. Smith. 2011. A discriminative model for joint morphological disambiguation and dependency parsing. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 885–894.
- Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011. Joint models for chinese pos tagging and dependency parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1180–1191.

- Ryan McDonald. 2006. *Discriminative Training and Spanning Tree Algorithms for Dependency Parsing. University of Pennsylvania*. Ph.D. thesis, PhD Thesis.
- 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, Marco Kuhlmann, and Johan Hall. 2009. An improved oracle for dependency parsing with online reordering. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 73–76.
- Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In Gertjan Van Noord, editor, *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.
- Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In Frank Keller, Stephen Clark, Matthew Crocker, and Mark Steedman, editors, *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together (ACL)*, pages 50–57.
- 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. 2009. Non-projective dependency parsing in expected linear time. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 351–359.
- Francesco Sartorio, Giorgio Satta, and Joakim Nivre. 2013. A transition-based dependency parser using a dynamic parsing strategy. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 135–144.
- 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 Second Workshop on Treebanks and Linguistic Theories (TLT)*, pages 189–200.
- Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 562–571.
- Yue Zhang and Joakim Nivre. 2011. Transition-based parsing with rich non-local features. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 188–193.

- Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 1391–1400.