CS 4650/7650, Parsing algorithms

Jacob Eisenstein

October 3, 2013

CKY example

	she	eats	sushi	with	tuna
she					
eats					
sushi					
with					
tuna					

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$				
eats					
sushi					
with					
tuna					

$$t[0,1,NP] = P(\mathrm{NP} \to \mathrm{she})$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0, 1, NP]$				
eats		$t[1, 2, VP]$			
sushi			$t[2, 3, NP]$		
with				$t[3, 4, P]$	
tuna					$t[4, 5, NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$			
eats		$t[1,2,VP]$			
sushi			$t[2,3,NP]$		
with				$t[3,4,P]$	
tuna					$t[4,5,NP]$

$$t[0,2,S] = P(\mathrm{S} \to \mathrm{NP\ VP}) \otimes t[0,1,NP] \otimes t[1,2,VP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$			
eats		$t[1,2,VP]$	$t[1,3,VP]$		
sushi			$t[2,3,NP]$		
with				$t[3,4,P]$	
tuna					$t[4,5,NP]$

$$t[1,3,VP] = P(\mathrm{VP} \rightarrow \mathrm{VP\ NP}) \otimes t[1,2,VP] \otimes t[2,3,NP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0, 1, NP]$	$t[0, 2, S]$			
eats		$t[1, 2, VP]$	$t[1, 3, VP]$		
sushi			$t[2, 3, NP]$	$\varnothing$	
with				$t[3, 4, P]$	
tuna					$t[4, 5, NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$			
eats		$t[1,2,VP]$	$t[1,3,VP]$		
sushi			$t[2,3,NP]$	$\varnothing$	
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$		
eats		$t[1,2,VP]$	$t[1,3,VP]$		
sushi			$t[2,3,NP]$	$\varnothing$	
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[0,3,S] = P(\mathrm{S} \rightarrow \mathrm{NP\ \ VP}) \otimes t[0,1,NP] \otimes t[1,3,VP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$		
eats		$t[1,2,VP]$	$t[1,3,VP]$		
sushi			$t[2,3,NP]$	$\varnothing$	
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$		
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	
sushi			$t[2,3,NP]$	$\varnothing$	
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0, 1, NP]$	$t[0, 2, S]$	$t[0, 3, S]$		
eats		$t[1, 2, VP]$	$t[1, 3, VP]$	$\varnothing$	
sushi			$t[2, 3, NP]$	$\varnothing$	
with				$t[3, 4, P]$	$t[3, 5, PP]$
tuna					$t[4, 5, NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$		
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[2,5,NP] = P(\mathrm{NP} \to \mathrm{NP\ PP}) \otimes t[2,3,NP] \otimes t[3,5,PP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$		
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[1,5,VP] = (P(\mathrm{VP} \rightarrow \mathrm{VP\ NP}) \otimes t[1,2,VP] \otimes t[2,5,NP])$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$
\begin{aligned}
t[1,5,VP] = &(P(\mathrm{VP} \rightarrow \mathrm{VP\ NP}) \otimes t[1,2,VP] \otimes t[2,5,NP]) \\
&\oplus (P(\mathrm{VP} \rightarrow \mathrm{VP\ PP}) \otimes t[1,3,VP] \otimes t[3,5,PP])
\end{aligned}
$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[0,5,S] = P(\mathrm{S} \to \mathrm{NP\ VP}) \otimes t[0,1,NP] \otimes t[1,5,VP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[0,5,S] = P(\mathrm{S} \rightarrow \mathrm{NP\ \ VP}) \otimes t[0,1,NP] \otimes t[1,5,VP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[0,5,S] = P(\mathrm{S} \rightarrow \mathrm{NP\ \ VP}) \otimes t[0,1,NP] \otimes t[1,5,VP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$ —	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$ -	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$ -	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[0,5,S] = P(\mathrm{S} \rightarrow \mathrm{NP}\ \mathrm{VP}) \otimes t[0,1,NP] \otimes t[1,5,VP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$t[0,5,S] = P(\mathrm{S} \rightarrow \mathrm{NP\ \ VP}) \otimes t[0,1,NP] \otimes t[1,5,VP]$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$
\begin{aligned}
t[0,5,S] = & P(\mathrm{S} \rightarrow \mathrm{NP\ \ VP}) \otimes P(\mathrm{NP} \rightarrow \mathrm{she}) \\
& \otimes P(\mathrm{VP} \rightarrow \mathrm{VP\ \ NP}) \otimes t[1,2,VP] \otimes t[2,5,NP]
\end{aligned}
$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$
\begin{aligned}
t[0,5,S] = & P(\mathrm{S} \to \mathrm{NP\ \ VP}) \otimes P(\mathrm{NP} \to \mathrm{she}) \\
& \otimes P(\mathrm{VP} \to \mathrm{VP\ \ NP}) \otimes P(\mathrm{VP} \to \mathrm{eats}) \otimes P(\mathrm{NP} \to \mathrm{NP\ \ PP}) \\
& \otimes t[2,3,NP] \otimes t[3,5,PP]
\end{aligned}
$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$
\begin{aligned}
t[0,5,S] = {} & P(\mathrm{S} \to \mathrm{NP\ \ VP}) \otimes P(\mathrm{NP} \to \mathrm{she}) \\
& \otimes P(\mathrm{VP} \to \mathrm{VP\ \ NP}) \otimes P(\mathrm{VP} \to \mathrm{eats}) \otimes P(\mathrm{NP} \to \mathrm{NP\ \ PP}) \\
& \otimes P(\mathrm{NP} \to \mathrm{sushi}) \\
& \otimes P(\mathrm{PP} \to \mathrm{P\ \ NP}) \otimes t[3,4,P] \otimes t[4,5,NP]
\end{aligned}
$$

CKY example

	she	eats	sushi	with	tuna
she	$t[0,1,NP]$	$t[0,2,S]$	$t[0,3,S]$	$\varnothing$	$t[0,5,S]$
eats		$t[1,2,VP]$	$t[1,3,VP]$	$\varnothing$	$t[1,5,VP]$
sushi			$t[2,3,NP]$	$\varnothing$	$t[2,5,NP]$
with				$t[3,4,P]$	$t[3,5,PP]$
tuna					$t[4,5,NP]$

$$
\begin{aligned}
t[0,5,S] = & P(\mathrm{S} \to \mathrm{NP\ \ VP}) \otimes P(\mathrm{NP} \to \mathrm{she}) \\
& \otimes P(\mathrm{VP} \to \mathrm{VP\ \ NP}) \otimes P(\mathrm{VP} \to \mathrm{eats}) \otimes P(\mathrm{NP} \to \mathrm{NP\ \ PP}) \\
& \otimes P(\mathrm{NP} \to \mathrm{sushi}) \\
& \otimes P(\mathrm{PP} \to \mathrm{P\ \ NP}) \otimes P(\mathrm{P} \to \mathrm{with}) \otimes P(\mathrm{NP} \to \mathrm{tuna})
\end{aligned}
$$

CKY example

Some important details about the example on the previous slide:

- The cells in the CKY table can hold more than one non-terminal.
  - For example, we could have $t[i, j, NP]$ and $t[i, j, VP]$.
  - The example was constructed to avoid this, to make it easier to show.
- In the final computation, we implicitly assume $\oplus = \max$ or $\vee$. If $\oplus = +$, we are adding probabilities, and we would need to consider both ways of deriving $\mathbf{x}_{1:5}$ from VP.

Shift-reduce example

move	stack	input
init	$\{\}$	She eats sushi with chopsticks

Shift-reduce example

move	**stack**	**input**
init	$\{\}$	She eats sushi with chopsticks
shift	$\{$ She $\}$	eats sushi with chopsticks

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks
reduce	{ NP, V, NP }	with chopsticks

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks
reduce	{ NP, V, NP }	with chopsticks
reduce	{ NP, VP }	with chopsticks

Shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks
reduce	{ NP, V, NP }	with chopsticks
reduce	{ NP, VP }	with chopsticks
S,R	{ NP, VP, P }	chopsticks

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks
reduce	{ NP, V, NP }	with chopsticks
reduce	{ NP, VP }	with chopsticks
S,R	{ NP, VP, P }	chopsticks
S,R	{ NP, VP, P, NP }	

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks
reduce	{ NP, V, NP }	with chopsticks
reduce	{ NP, VP }	with chopsticks
S,R	{ NP, VP, P }	chopsticks
S,R	{ NP, VP, P, NP }	
reduce	{ NP, VP, PP }	

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks
reduce	{ NP, V, NP }	with chopsticks
reduce	{ NP, VP }	with chopsticks
S,R	{ NP, VP, P }	chopsticks
S,R	{ NP, VP, P, NP }	
reduce	{ NP, VP, PP }	
reduce	{ NP, VP }	

Shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats, }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
shift	{ NP, V, sushi }	with chopsticks
reduce	{ NP, V, NP }	with chopsticks
reduce	{ NP, VP }	with chopsticks
S,R	{ NP, VP, P }	chopsticks
S,R	{ NP, VP, P, NP }	
reduce	{ NP, VP, PP }	
reduce	{ NP, VP }	
reduce	{ S }	

Failed shift-reduce example

move	stack	input
init	$\{\}$	She eats sushi with chopsticks

Failed shift-reduce example

move	stack	input
init	$\{\}$	She eats sushi with chopsticks
shift	$\{$ She $\}$	eats sushi with chopsticks

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks

Failed shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks
shift	{ S, sushi }	with chopsticks

Failed shift-reduce example

move	stack	input
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks
shift	{ S, sushi }	with chopsticks
reduce	{ S, NP }	with chopsticks

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks
shift	{ S, sushi }	with chopsticks
reduce	{ S, NP }	with chopsticks
S,R	{ S, NP, P }	chopsticks

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks
shift	{ S, sushi }	with chopsticks
reduce	{ S, NP }	with chopsticks
S,R	{ S, NP, P }	chopsticks
S,R	{ S, NP, P, NP }	

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks
shift	{ S, sushi }	with chopsticks
reduce	{ S, NP }	with chopsticks
S,R	{ S, NP, P }	chopsticks
S,R	{ S, NP, P, NP }	
R	{ S, NP, PP }	

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks
shift	{ S, sushi }	with chopsticks
reduce	{ S, NP }	with chopsticks
S,R	{ S, NP, P }	chopsticks
S,R	{ S, NP, P, NP }	
R	{ S, NP, PP }	
R	{ S, NP }	

Failed shift-reduce example

move	**stack**	**input**
init	{}	She eats sushi with chopsticks
shift	{ She }	eats sushi with chopsticks
reduce	{ NP }	eats sushi with chopsticks
shift	{ NP, eats }	sushi with chopsticks
reduce	{ NP, V }	sushi with chopsticks
reduce	{ S }	sushi with chopsticks
shift	{ S, sushi }	with chopsticks
reduce	{ S, NP }	with chopsticks
S,R	{ S, NP, P }	chopsticks
S,R	{ S, NP, P, NP }	
R	{ S, NP, PP }	
R	{ S, NP }	

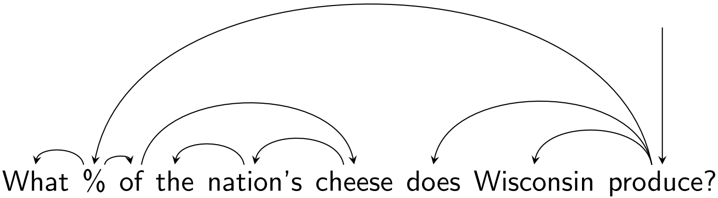
We're stuck! Now we have to **backtrack**.

Dependency parsing in action

Dependency parsing is used in many real-world applications, like question answering (Cui et al, 2005):

Dependency parsing in action

Dependency parsing is used in many real-world applications, like question answering (Cui et al, 2005):



Dependency parsing in action

Dependency parsing is used in many real-world applications, like question answering (Cui et al, 2005):

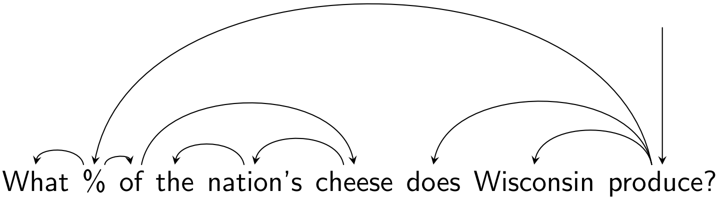
What % of the nation's cheese does Wisconsin produce?

In Wisconsin, where farmers produce 28 % of the nation's cheese, . . .

Dependency parsing in action

Question answering works by searching for statements which match well against the query.

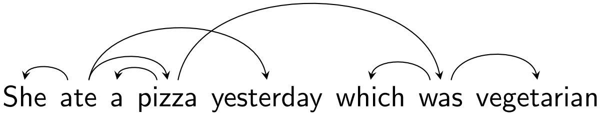
- In the surface form of the question, $\mathrm{produce}$ and $\%$ are six words apart.
- But in the dependency parse, they're adjacent.



Projectivity

In **projective** dependency parsing, there are no crossing edges.

- Crossing edges are rare in English:

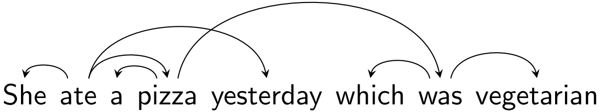


[1] figure from (Nivre 2007)

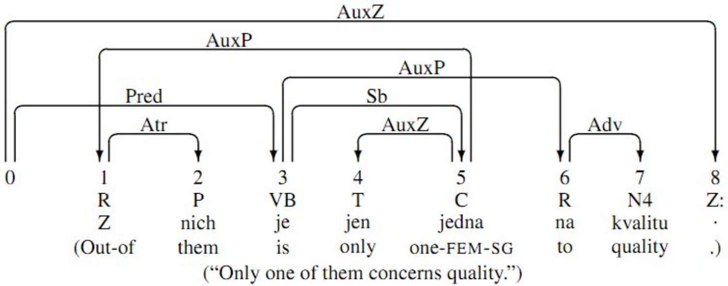
Projectivity

In **projective** dependency parsing, there are no crossing edges.

- Crossing edges are rare in English:



- They are more common in other languages, like Czech:[1]



[1]figure from (Nivre 2007)

Projective dependency parsing

- The **Eisner** algorithm is similar to CKY.
- But it keeps four tables instead of 1!
  - scores of **incomplete** subtrees from i to j, headed to the left
  - scores of **incomplete** subtrees from i to j, headed to the right
  - scores of **complete** subtrees from i to j, headed to the left
  - scores of **complete** subtrees from i to j, headed to the right
- Our goal is to produce a complete tree from 0 to N.
- Complete subtrees can combine into an incomplete tree, if they are heading in opposite directions.
  - E.g. $[a \rightarrow b]$ $[c \leftarrow d]$ becomes $([a \rightarrow b] \rightarrow [c \leftarrow d])$,
  - where square brackets like $[a \rightarrow b]$ indicate a complete tree
  - and round brackets like $(x \rightarrow y)$ indicate an incomplete tree
- Incomplete subtrees can subsume complete subtrees heading in the same direction.
  - E.g. $(a \rightarrow b)[c \rightarrow d]$ becomes $[[a \rightarrow b] \rightarrow [c \rightarrow d]]$
- See my "Eisner Algorithm Worksheet" for the specific recurrences.

Projective dependency parsing: The Eisner Algorithm

- ROOT She gave him the ball
- ROOT (She $\leftarrow$ gave) (gave $\rightarrow$ him) (the $\leftarrow$ ball)
- [ROOT] [She $\leftarrow$ gave] [gave $\rightarrow$ him] [the $\leftarrow$ ball]

ROOT she gave him the ball

Projective dependency parsing: The Eisner Algorithm

- ROOT She gave him the ball
- ROOT (She $\leftarrow$ gave) (gave $\rightarrow$ him) (the $\leftarrow$ ball)
- [ROOT] [She $\leftarrow$ gave] [gave $\rightarrow$ him] [the $\leftarrow$ ball]

ROOT she gave him the ball

- ([ROOT] $\rightarrow$ [She $\leftarrow$ gave]) ([gave $\rightarrow$ him] $\rightarrow$ [the $\leftarrow$ ball])

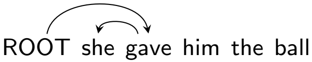
ROOT she gave him the ball

ROOT she gave him the ball

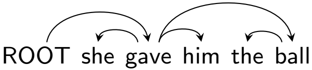
Projective dependency parsing: The Eisner Algorithm

- ROOT She gave him the ball
- ROOT (She ← gave) (gave → him) (the ← ball)
- [ROOT] [She ← gave] [gave → him] [the ← ball]

- ([ROOT] → [She ← gave]) ([gave → him] → [the ← ball])



- ([ROOT] → [She ← gave]) [[gave → him] → [the ← ball]]
- [([ROOT] → [She ← gave]) → [[gave → him] → [the ← ball]]]



The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

incomplete

$\rightarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

complete

$\leftarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

complete

$\rightarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

complete

$\leftarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

incomplete

$\rightarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

complete

$\rightarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

complete

$\leftarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

complete

$\rightarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

complete

$\rightarrow$	plastic	cup	holders
ROOT			
plastic			
cup			

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

complete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	4
cup			4

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

complete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	4
cup			4

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$		
plastic		2	
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

complete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	4
cup			4

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	6
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

complete

$\rightarrow$	plastic	cup	holders
ROOT	1		
plastic		-1	
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	4
cup			4

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	6
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

complete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	$-\infty$
plastic		2	4
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	6
cup			4

complete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	$-\infty$
plastic		2	4
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	7
plastic		-1	3
cup			-1

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	
plastic		2	6
cup			4

complete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	$-\infty$
plastic		2	4
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	7
plastic		-1	3
cup			-1

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	$-\infty$
plastic		2	6
cup			4

complete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	
plastic		-1	3
cup			-1

The Eisner Algorithm

Edge weights:

	ROOT	plastic	cup	holders
ROOT		1	1	1
plastic	$-\infty$		-1	-1
cup	$-\infty$	2		-1
holders	$-\infty$	0	4	

incomplete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	$-\infty$
plastic		2	4
cup			4

complete

$\leftarrow$	plastic	cup	holders
ROOT	$-\infty$	$-\infty$	$-\infty$
plastic		2	6
cup			4

incomplete

$\rightarrow$	plastic	cup	holders
ROOT	1	3	7
plastic		-1	3
cup			-1

complete

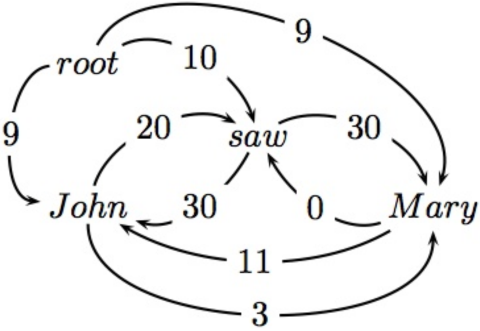
$\rightarrow$	plastic	cup	holders
ROOT	1	3	7
plastic		-1	3
cup			-1

Arc-factored dependency parsing algorithms

- The Eisner algorithm produces projective dependency parses.
- Non-projective dependency parsing can be viewed as maximum spanning tree.
- A slightly modified version of Chu-Liu-Edmonds can then be applied.
- The next few slides are from Joakim Nivre and Ryan McDonald.

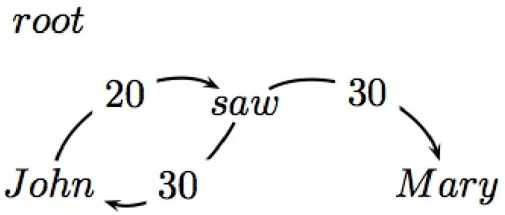
Chu-Liu-Edmonds

- x = root John saw Mary



Chu-Liu-Edmonds

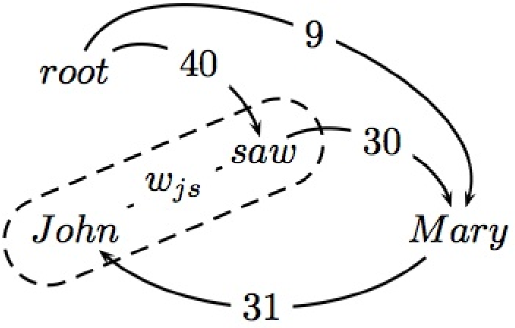
- Find highest scoring incoming arc for each vertex



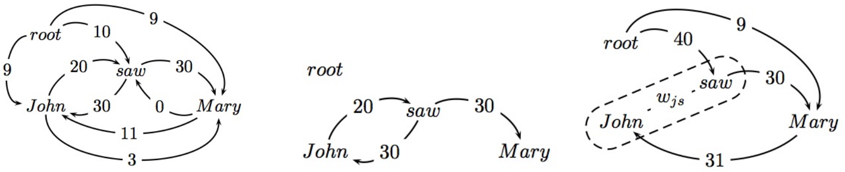
- If this is a tree, then we have found MST!!

Chu-Liu-Edmonds

- If not a tree, identify cycle and contract
- Recalculate arc weights into and out-of cycle

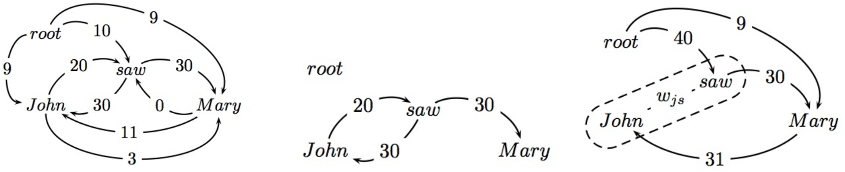


Chu-Liu-Edmonds



- Outgoing arc weights
  - Equal to the max of outgoing arc over all vertexes in cycle
  - e.g., John $\rightarrow$ Mary is 3 and saw $\rightarrow$ Mary is 30

Chu-Liu-Edmonds

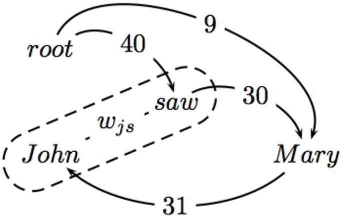


- Incoming arc weights
  - Equal to the weight of best spanning tree that includes head of incoming arc, and all nodes in cycle
  - root $\rightarrow$ saw $\rightarrow$ John is 40 (**)
  - root $\rightarrow$ John $\rightarrow$ saw is 29

Chu-Liu-Edmonds

Theorem

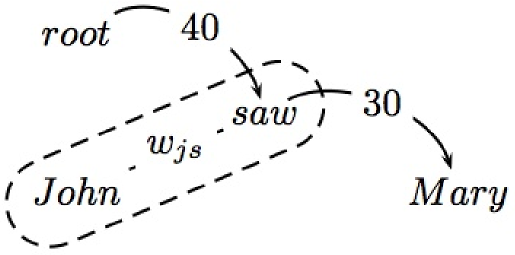
The weight of the MST of this contracted graph is equal to the weight of the MST for the original graph



- Therefore, recursively call algorithm on new graph

Chu-Liu-Edmonds

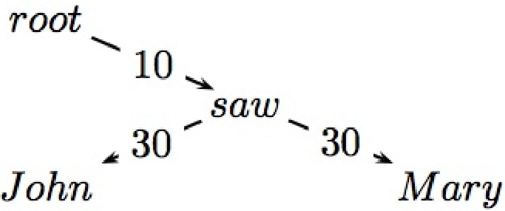
- This is a tree and the MST for the contracted graph!!



- Go back up recursive call and reconstruct final graph

Chu-Liu-Edmonds

- This is the MST!!



Chu-Liu-Edmonds Code

Chu-Liu-Edmonds(G_x, w)

1. Let $M = \{(i^*, j) : j \in V_x, i^* = \arg\max_{i'} w_{ij}\}$
2. Let $G_M = (V_x, M)$
3. If G_M has no cycles, then it is an MST: return G_M
4. Otherwise, find a cycle C in G_M
5. Let $< G_C, c, ma >=$ contract(G, C, w)
6. Let $G =$ Chu-Liu-Edmonds(G_C, w)
7. Find vertex $i \in C$ such that $(i', c) \in G$ and $ma(i', c) = i$
8. Find arc $(i'', i) \in C$
9. Find all arc $(c, i''') \in G$
10. $G = G \cup \{(ma(c, i'''), i''')\}_{\forall (c,i''') \in G} \cup C \cup \{(i', i)\} - \{(i'', i)\}$
11. Remove all vertices and arcs in G containing c
12. return G

- Reminder: $w_{ij} = \arg\max_k w_{ij}^k$

Summary of dependency parsing algorithms

- The Eisner algorithm for projective dependency parsing is $\mathcal{O}(N^3)$
- MST for non-projective dependency parsing is also $\mathcal{O}(N^3)$, but Tarjan's algorithm is $\mathcal{O}(N^2)$.
- We can also apply shift-reduce to dependency parsing, with complexity $\mathcal{O}(N)$ – but we're not guaranteed to get the best-scoring parse.
- All of these algorithms depend on the arc-factoring assumption:

$$\psi(G, \mathbf{x}) = \sum_{\langle i \to j \rangle \in G} \psi(i \to j, \mathbf{x})$$

  where $\psi(i \to j, \mathbf{x})$ can be a log-probability or an inner productive of weights and features.
- **Second-order** dependency parsing relaxes the arc-factoring assumption.