

2018/11/23 22:00 修正

計算言語学₇

構造分類 (統計的構文解析)

東京大学生産技術研究所

吉永 直樹

site: <http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/class/cl/>

言語をカテゴリ(クラス)の構造に分類する

品詞タグ付け (part-of-speech tagging)

Lucy	in	the	sky	with	diamonds
NNP	IN	DT	NN	IN	NNS

単語に対する品詞の系列

固有表現認識 (named entity recognition)

Ringo	Star	has joined	the	Beatles
PERSON	Non-NE			ORGANIZATION

固有表現のチャンクとその分類

単語分割 (word segmentation)

中	国	人	参	政	权
0	1	1	0	0	1

単語境界の有無の系列

依存構造解析 (dependency parsing)

Ringo Star has joined the Beatles

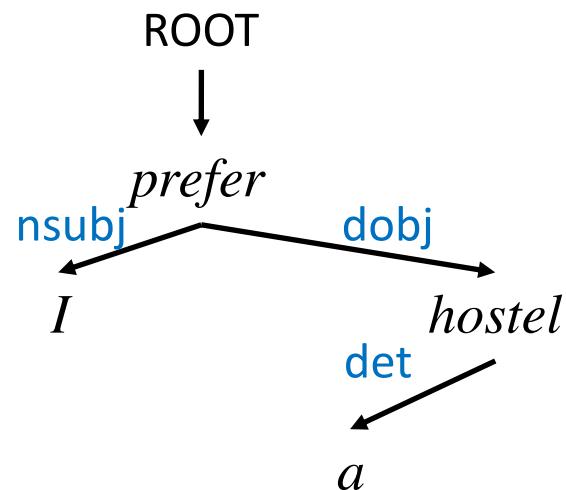
The diagram shows six words: Ringo, Star, has, joined, the, and Beatles. Blue curved arrows point from each word except 'joined' to the word 'joined'. This indicates that 'joined' is the root node or head of the sentence, and all other words depend on it.

依存構造木 or 結合操作の系列

個々のラベルの間に依存関係がある

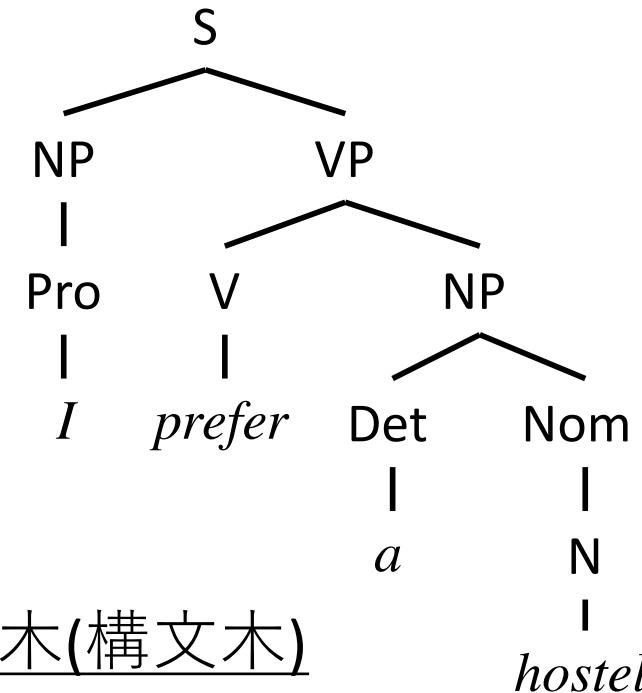
構文構造を記述する二つのアプローチ

- 依存文法 [Teschnière 1959]
 - 単語間の統語的依存関係を解析
 - 日本語やチェコ語など、語順が自由な言語で発達
 - 統計的手法



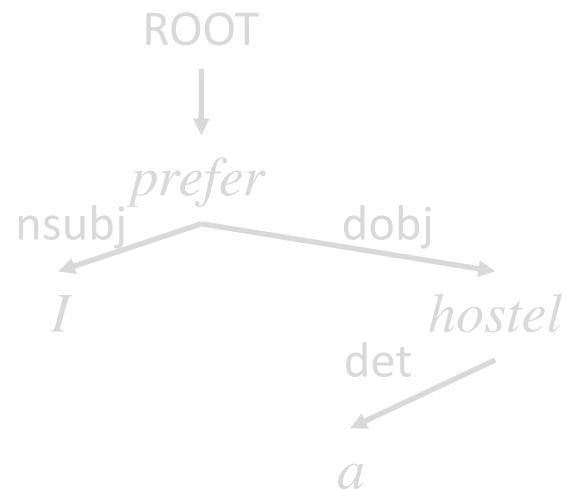
11/23/18

- 句構造文法 [Chomsky 1956]
 - 単語列(構成素)の階層的包含関係を解析
 - 英語など、語順が比較的固定された言語で発達
 - 形式文法 + 統計的手法



構文構造を記述する二つのアプローチ

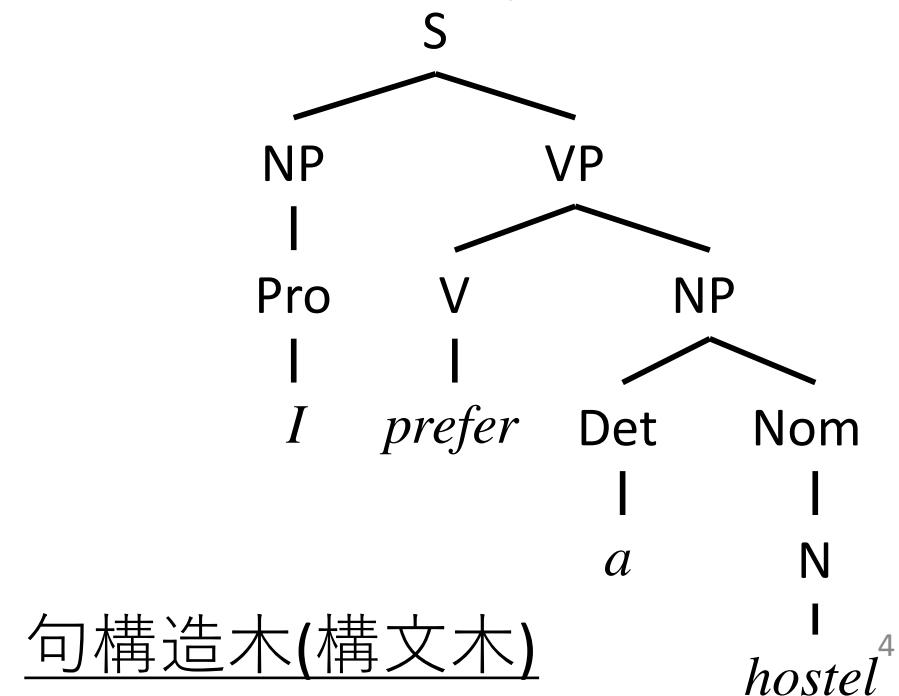
- 依存文法 [Tesiⁿière 1959]
 - 単語間の統語的依存関係を解析
 - 日本語やチェコ語など、語順が自由な言語で発達
 - 統計的手法



11/23/18

依存構造木

- 句構造文法 [Chomsky 1956]
 - 単語列(構成素)の階層的包含関係を解析
 - 英語など、語順が比較的固定された言語で発達
 - 形式文法 + 統計的手法



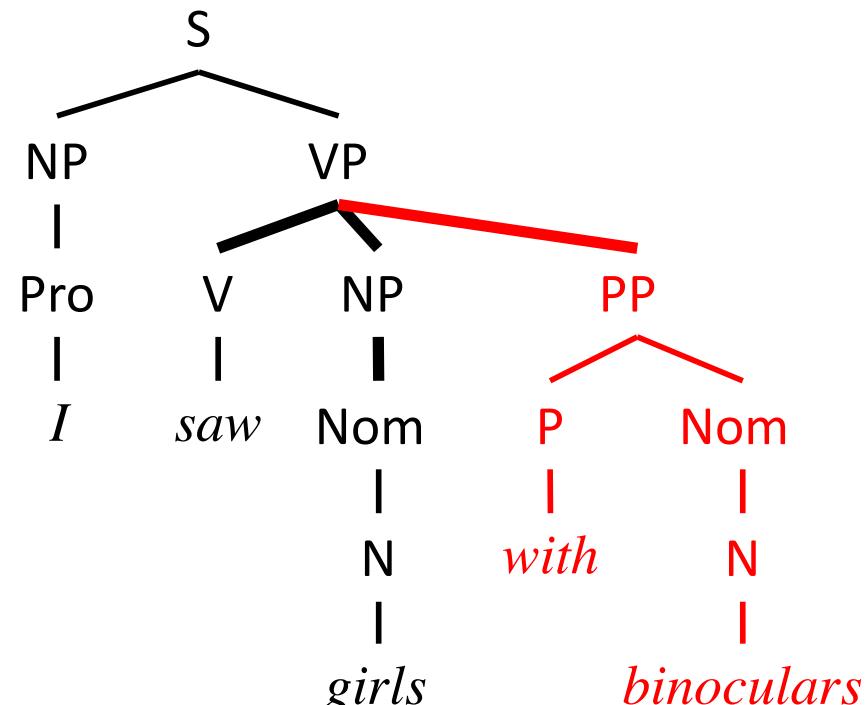
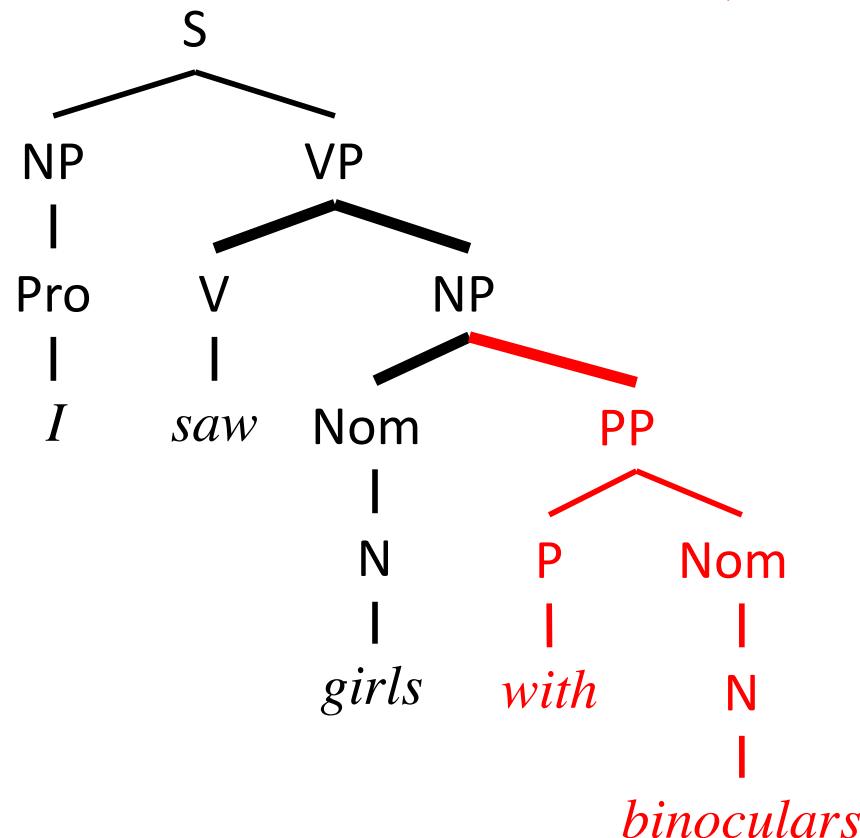
句構造木(構文木)

hostel⁴

部分構造の曖昧性 (1/2): Attachment ambiguity

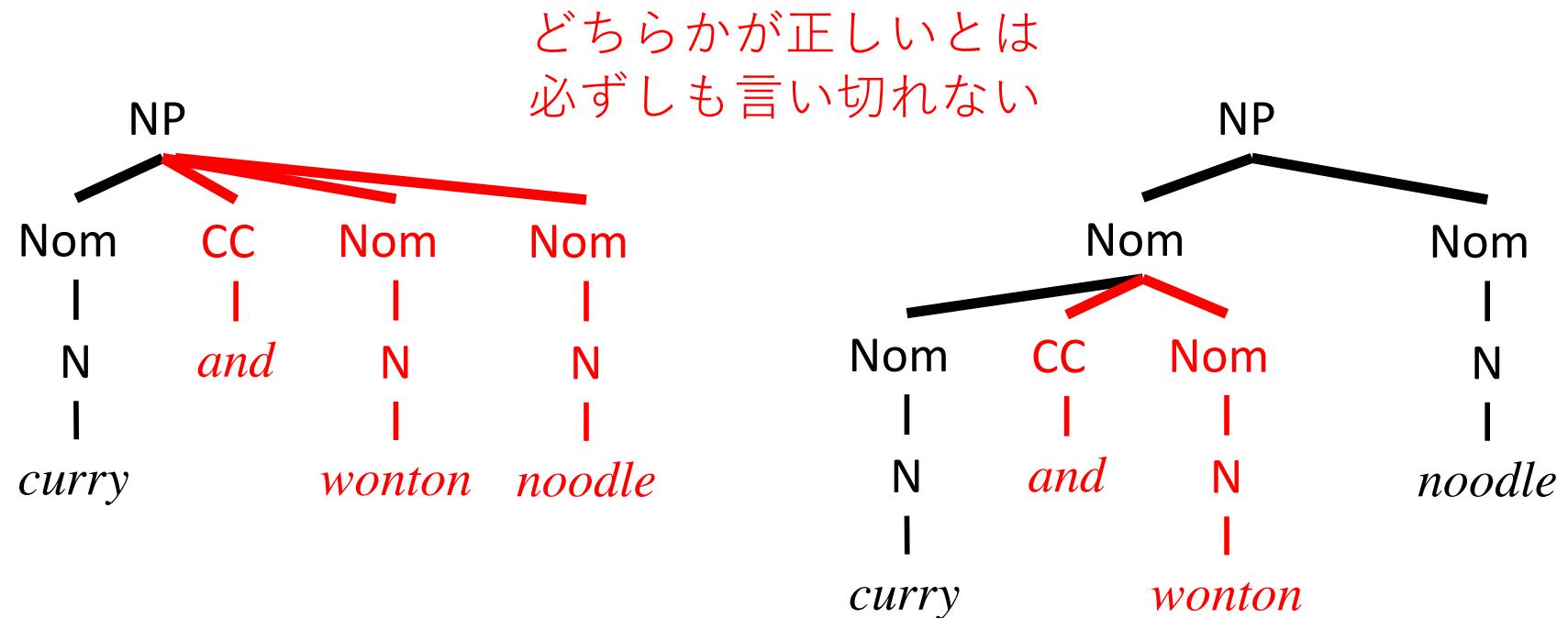
- 句が修飾する句の候補の曖昧性

どちらかが正しいとは
必ずしも言い切れない



部分構造の曖昧性 (1/2): Coordination ambiguity

- 並列句の並列範囲の曖昧性



部分構造の曖昧性をどのようにモデル化するか？

Probabilistic Context-Free Grammar (PCFG)

確率的文脈自由文法

- アイデア: 構文木の部分構造に確率を付与し,
構文木の確率を部分構造の確率の積で計算

$$P(T) = \prod_{A \rightarrow \beta \in T} P(\beta|A)$$

- 確率的文脈自由文法(PCFG)

- N : 非終端記号 (変数) 構成素のラベル
- Σ : 終端記号 語彙(単語)
- R : 生成規則 $A \rightarrow \beta[p]$ ($A \in N, \beta \in (\Sigma \cup N)^*, p = P(\beta|A)$)
 構成素間の包含関係とその確率
- S : 開始記号 ($S \in N$) 文

$$\sum_{\beta} P(\beta|A) = 1$$

PCFG を用いた曖昧性解消

- 文 S に対し, PCFG で計算される構文木の確率 $P(T)$ に基づき構文木候補から正解の構文木を選択

終端文字列として S を生成する構文木 T

$$\hat{T}(S) = \operatorname{argmax}_{T:S=\text{yield}(T)} P(T|S)$$

$$= \operatorname{argmax}_{T:S=\text{yield}(T)} \frac{P(T)P(S|T)}{P(S)}$$

構文木 T は文 S を含むので 1

$$= \operatorname{argmax}_{T:S=\text{yield}(T)} P(T)$$

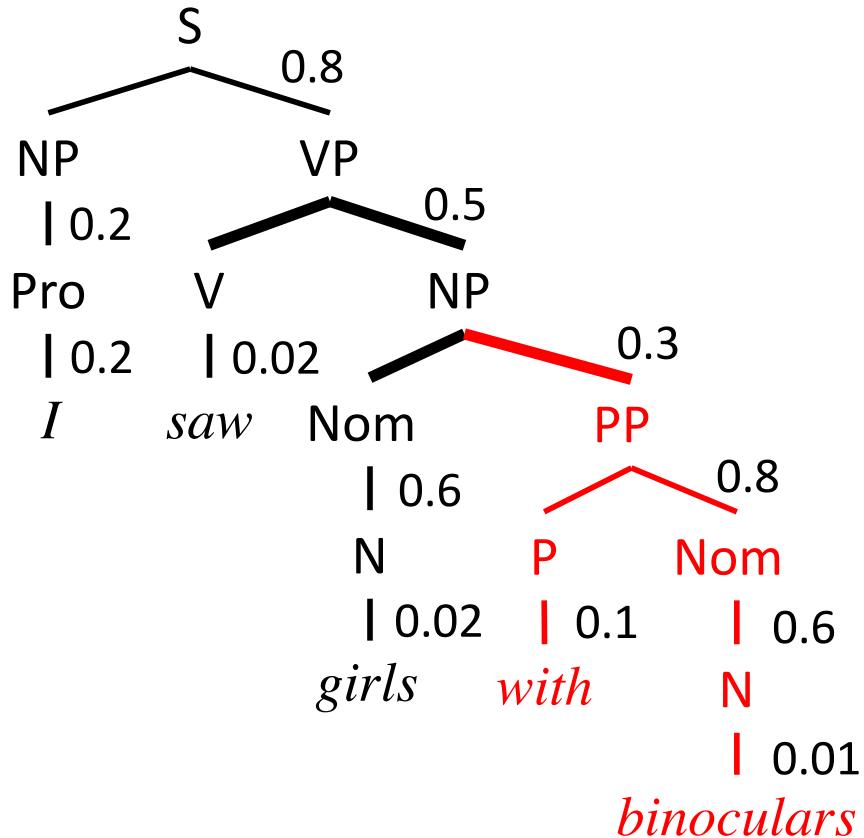
$$= \operatorname{argmax}_{T:S=\text{yield}(T)} \prod_{A \rightarrow \beta \in T} P(\beta|A)$$

- 言語モデルに用い単語の予測を行うこともできる

The contract_{NP} ended_V with a loss of 7 cents_{PP} after trading as low as 9 cents

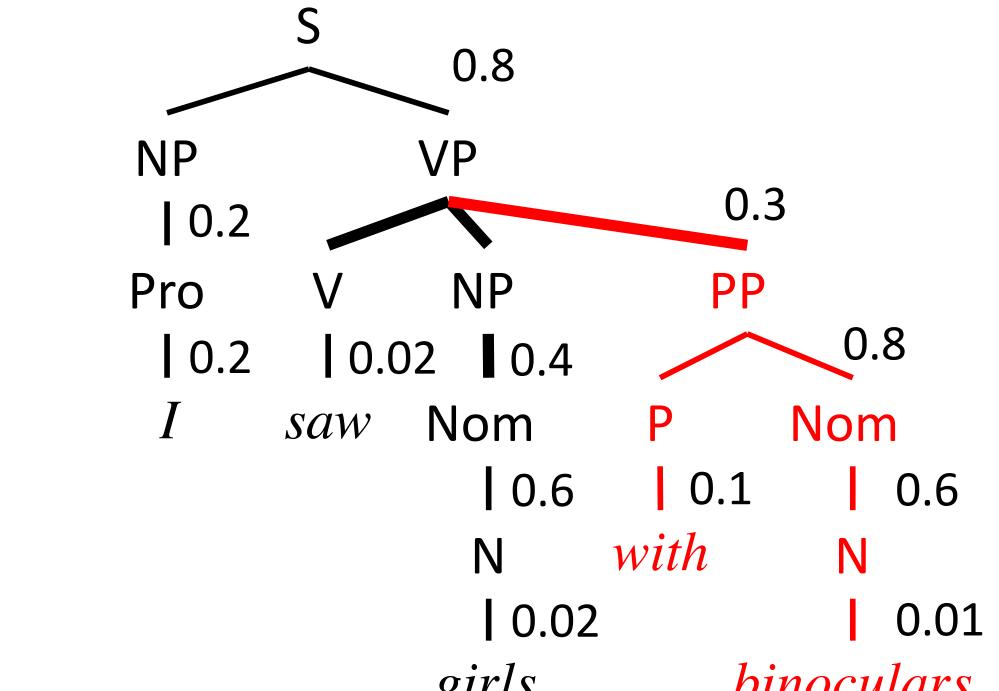
PCFG を用いた曖昧性解消: 例

- *I saw girls with binoculars* に対する構文木



$$\begin{aligned}
 P(T) &= 0.8 \cdot 0.2 \cdot 0.2 \\
 &\quad \cdot 0.5 \cdot 0.3 \\
 &\quad \cdot 0.02 \cdot 0.6 \cdot 0.02 \cdot 0.8 \cdot 0.1 \cdot 0.6 \cdot 0.01 \\
 &= 5.5296 \cdot 10^{-10}
 \end{aligned}$$

11/23/18



$$\begin{aligned}
 P(T) &= 0.8 \cdot 0.2 \cdot 0.2 \\
 &\quad \cdot 0.3 \cdot 0.4 \\
 &\quad \cdot 0.02 \cdot 0.6 \cdot 0.02 \cdot 0.8 \cdot 0.1 \cdot 0.6 \cdot 0.01 \\
 &= 4.42368 \cdot 10^{-10}
 \end{aligned}$$

9

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキイ標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP [1.0]$

$VP \rightarrow V NP [0.7]$

$VP \rightarrow V NP PP [0.3]$

$PP \rightarrow P NP [1.0]$

$NP \rightarrow NP PP [0.5]$

$NP \rightarrow N [0.5]$

$S : S$

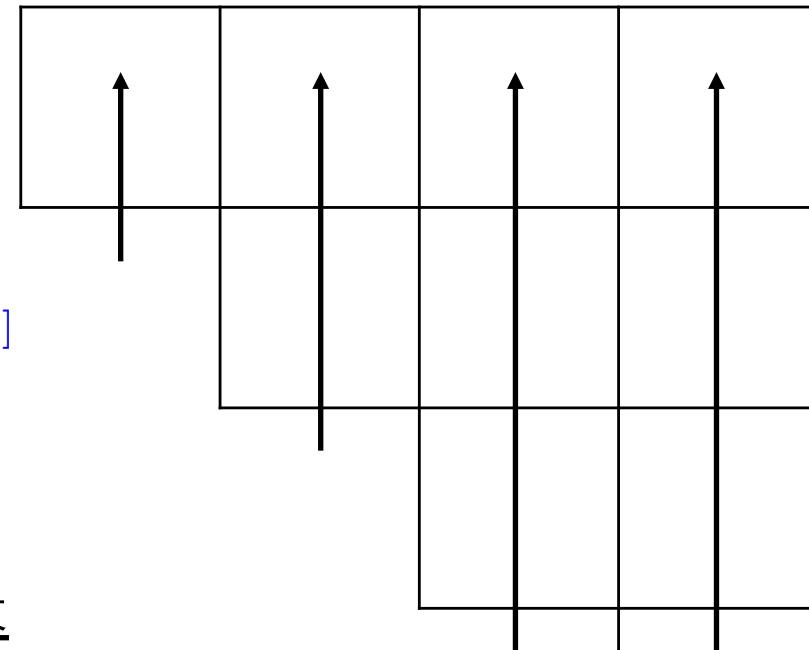
$V \rightarrow watch [1.0]$

$N \rightarrow girls [0.8]$

$N \rightarrow glasses [0.2]$

$P \rightarrow with [1.0]$

$_0 watch_1 girls_2 with_3 glasses_4$



CKY 表

各セル $[i, j]$ は部分木の
根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキイ標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

追加した非終端記号に関する規則の確率は1

$_0 watch_1 girls_2 with_3 glasses_4$

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V NP_{[0.7]}$

$VP \rightarrow V X_{[0.3]}$

$X \rightarrow NP PP_{[1.0]}$

$PP \rightarrow P NP_{[1.0]}$

$NP \rightarrow NP PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

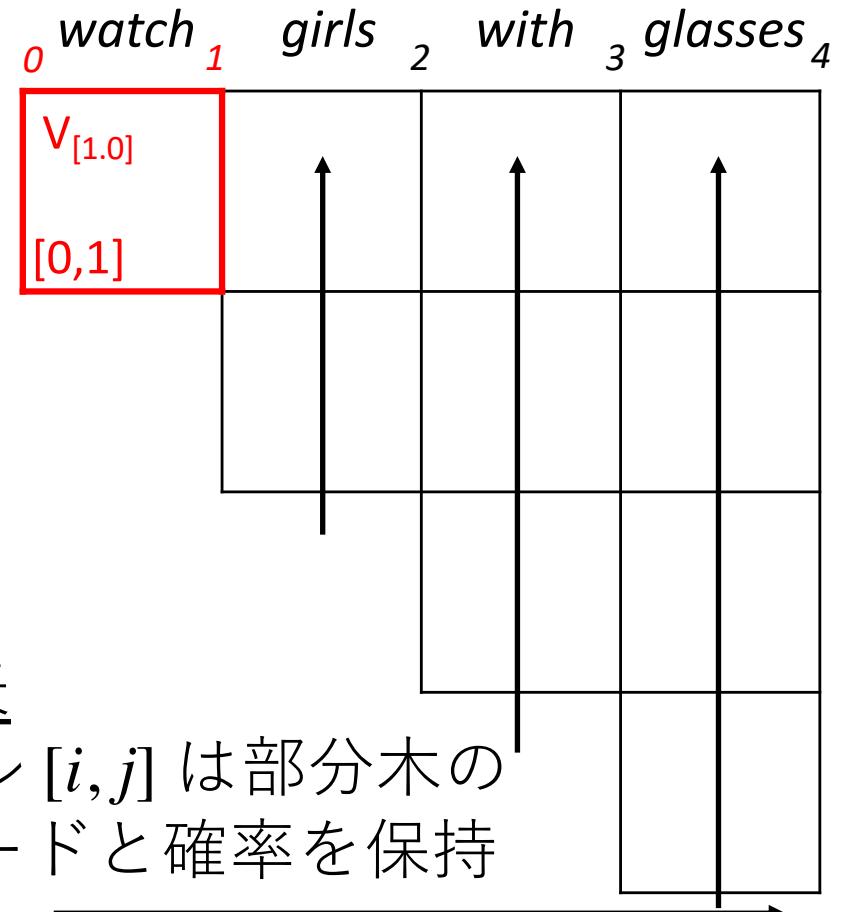
$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持



PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

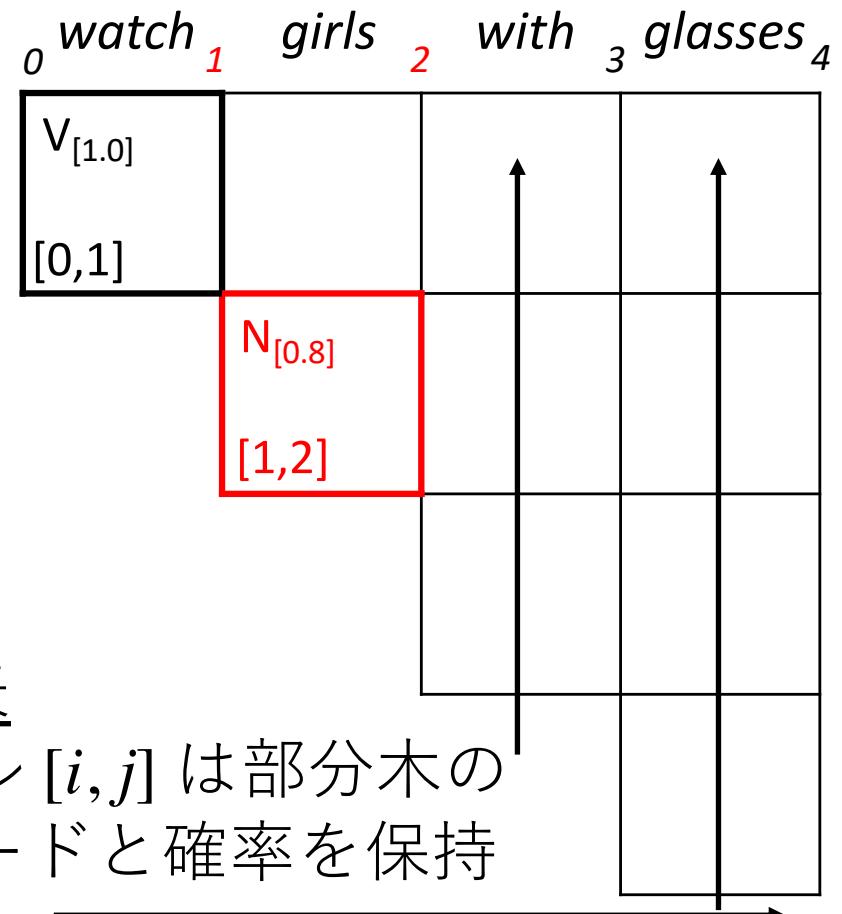
$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持



PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V NP_{[0.7]}$

$VP \rightarrow V X_{[0.3]}$

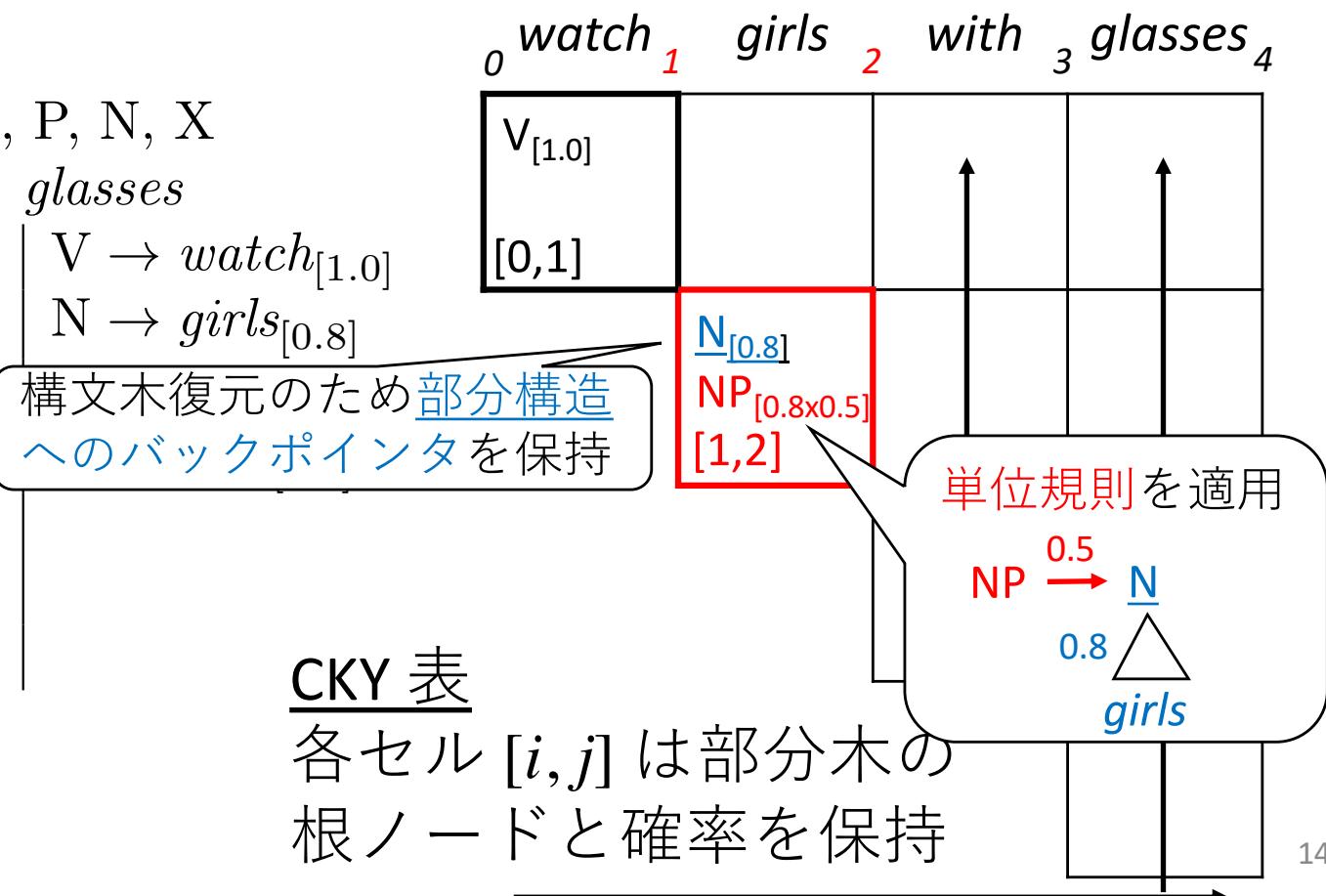
$X \rightarrow NP PP_{[1.0]}$

$PP \rightarrow P NP_{[1.0]}$

$NP \rightarrow NP PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$



PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法はチョムスキー標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

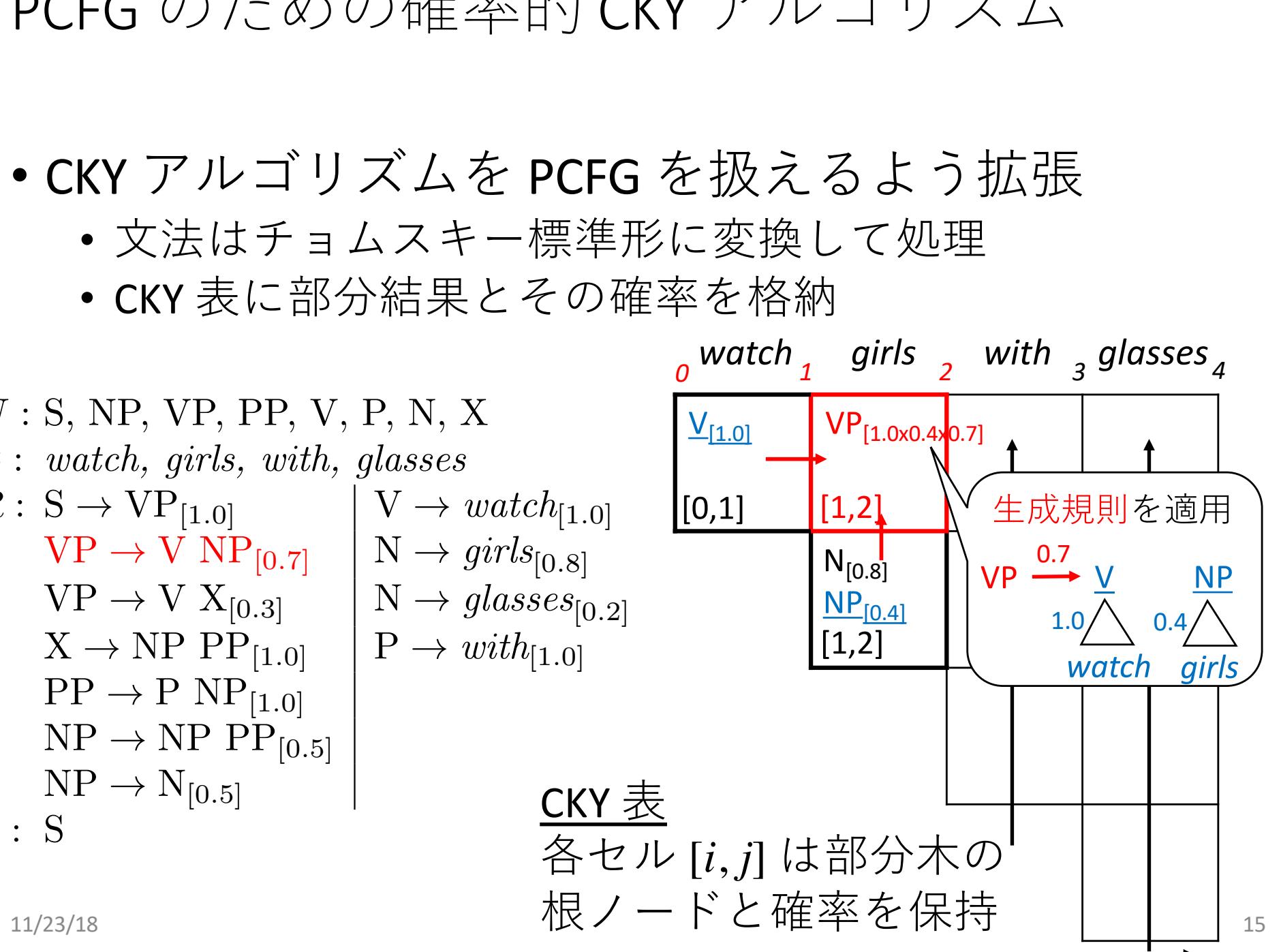
$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$



PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$
 $VP \rightarrow V\ NP_{[0.7]}$
 $VP \rightarrow V\ X_{[0.3]}$
 $X \rightarrow NP\ PP_{[1.0]}$
 $PP \rightarrow P\ NP_{[1.0]}$
 $NP \rightarrow NP\ PP_{[0.5]}$
 $NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$
 $N \rightarrow girls_{[0.8]}$
 $N \rightarrow glasses_{[0.2]}$
 $P \rightarrow with_{[1.0]}$

$V_{[1.0]}$ [0,1]	$VP_{[0.28]}$ [0,2]			
		$N_{[0.8]}$ $NP_{[0.4]}$ [1,2]		
			$P_{[1.0]}$ [2,3]	

CKY 表
各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$
 $VP \rightarrow V\ NP_{[0.7]}$
 $VP \rightarrow V\ X_{[0.3]}$
 $X \rightarrow NP\ PP_{[1.0]}$
 $PP \rightarrow P\ NP_{[1.0]}$
 $NP \rightarrow NP\ PP_{[0.5]}$
 $NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$
 $N \rightarrow girls_{[0.8]}$
 $N \rightarrow glasses_{[0.2]}$
 $P \rightarrow with_{[1.0]}$

$0 \ watch \ 1 \ girls \ 2 \ with \ 3 \ glasses \ 4$				
$V_{[1.0]}$	$VP_{[0.28]}$			
[0,1]	[0,2]			
		$N_{[0.8]}$ $NP_{[0.4]}$		
		[1,2]	[1,3]	
				$P_{[1.0]}$
				[2,3]

CKY 表
各セル $[i, j]$ は部分木の
根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキイ標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

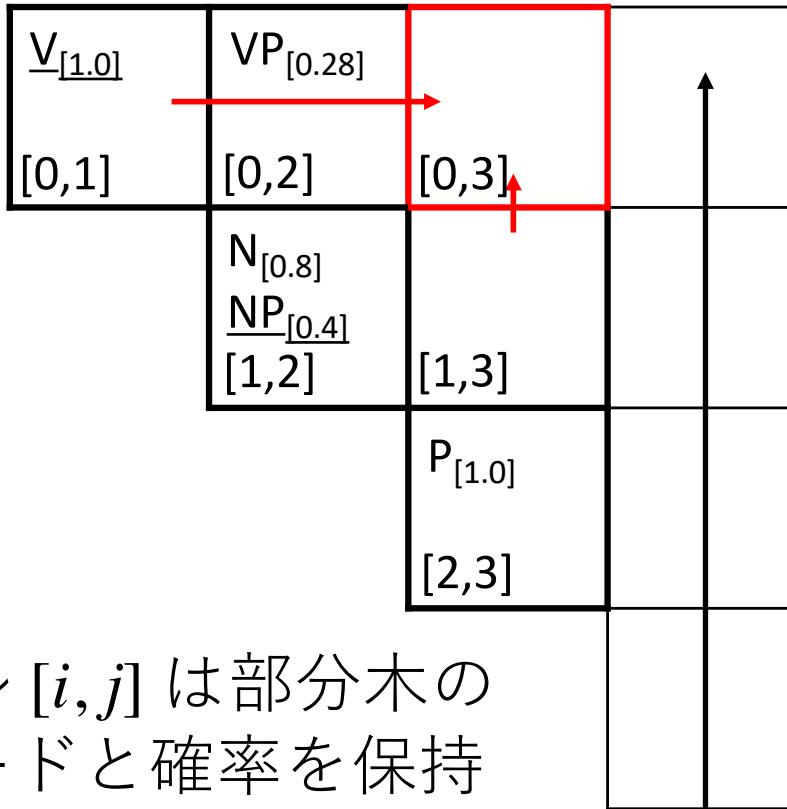
$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

		0 <i>watch</i> 1 <i>girls</i> 2 <i>with</i> 3 <i>glasses</i> 4		
V _[1.0]	VP _[0.28]			
	[0,1]	[0,2]	[0,3]	
	N _[0.8]			
	NP _[0.4]			
	[1,2]			
		P _[1.0]		
		[2,3]		

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持



PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

		0 <i>watch</i>	1 <i>girls</i>	2 <i>with</i>	3 <i>glasses</i>	4
$V_{[1.0]}$	$[0,1]$	$VP_{[0.28]}$	$[0,2]$	$[0,3]$		
				$N_{[0.8]}$		
				$NP_{[0.4]}$	$[1,3]$	
				$P_{[1.0]}$		
					$[2,3]$	

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$
 $VP \rightarrow V\ NP_{[0.7]}$
 $VP \rightarrow V\ X_{[0.3]}$
 $X \rightarrow NP\ PP_{[1.0]}$
 $PP \rightarrow P\ NP_{[1.0]}$
 $NP \rightarrow NP\ PP_{[0.5]}$
 $NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$
 $N \rightarrow girls_{[0.8]}$
 $N \rightarrow glasses_{[0.2]}$
 $P \rightarrow with_{[1.0]}$

	$watch_{[1.0]}$ [0,1]	$VP_{[0.28]}$ [0,2]		
			$N_{[0.8]}$ $NP_{[0.4]}$ [1,2]	
			$P_{[1.0]}$ [2,3]	
				$N_{[0.2]}$ $NP_{[0.2 \times 0.5]}$ [3,4]

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキイ 標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

N : S, NP, VP, PP, V, P, N, X

Σ : *watch, girls, with, glasses*

$R : S \rightarrow VP_{[1.0]}$
 $VP \rightarrow V NP_{[0.7]}$
 $VP \rightarrow V X_{[0.3]}$
 $X \rightarrow NP PP_{[1.0]}$
 $PP \rightarrow P NP_{[1.0]}$
 $NP \rightarrow NP PP_{[0.5]}$
 $NP \rightarrow N_{[0.5]}$
 $S : S$

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$
 $VP \rightarrow V\ NP_{[0.7]}$
 $VP \rightarrow V\ X_{[0.3]}$
 $X \rightarrow NP\ PP_{[1.0]}$
 $PP \rightarrow P\ NP_{[1.0]}$
 $NP \rightarrow NP\ PP_{[0.5]}$
 $NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$
 $N \rightarrow girls_{[0.8]}$
 $N \rightarrow glasses_{[0.2]}$
 $P \rightarrow with_{[1.0]}$

$watch_{[1.0]}$				$girls_{[0.8]}$	$with_{[1.0]}$	$glasses_{[0.2]}$
$[0,1]$	$[0,2]$	$[0,3]$	$[0,4]$			
$V_{[1.0]}$	$VP_{[0.28]}$					
$[0,1]$	$[0,2]$	$[0,3]$	$[0,4]$			
				$N_{[0.8]}$	$X_{[0.4 \times 0.1 \times 1.0]}$	
				$NP_{[0.4]}$	$NP_{[0.4 \times 0.1 \times 0.5]}$	
		$[1,2]$	$[1,3]$	$[1,2]$	$[1,4]$	
				$P_{[1.0]}$	$PP_{[0.1]}$	
				$[2,3]$	$[2,4]$	
				$N_{[0.2]}$	$NP_{[0.1]}$	
				$[3,4]$		

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$
 $VP \rightarrow V\ NP_{[0.7]}$
 $VP \rightarrow V\ X_{[0.3]}$
 $X \rightarrow NP\ PP_{[1.0]}$
 $PP \rightarrow P\ NP_{[1.0]}$
 $NP \rightarrow NP\ PP_{[0.5]}$
 $NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$
 $N \rightarrow girls_{[0.8]}$
 $N \rightarrow glasses_{[0.2]}$
 $P \rightarrow with_{[1.0]}$

		0 <i>watch</i>	1 <i>girls</i>	2 <i>with</i>	3 <i>glasses</i>	4
		$V_{[1.0]}$	$VP_{[0.28]}$			
i	j	$[0,1]$	$[0,2]$	$[0,3]$		
		$V_{[1.0]}$	$VP_{[0.28]}$			
$i=1$	$j=2$			$N_{[0.8]}$	$X_{[0.04]}$	
	$j=3$		$NP_{[0.4]}$	$[1,2]$	$NP_{[0.02]}$	$[1,4]$
$i=2$	$j=3$			$P_{[1.0]}$	$PP_{[0.1]}$	
	$j=4$			$[2,3]$	$[2,4]$	
$i=3$	$j=4$				$N_{[0.2]}$	
	$j=4$				$NP_{[0.1]}$	$[3,4]$

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法はチョムスキーモルヒニ形式に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

CKY 表				
各セル $[i, j]$ は部分木の根ノードと確率を保持				
0	$watch$	1	$girls$	2
$V_{[1.0]}$	$VP_{[0.28]}$			$VP_{[\max(1.0 \times 0.04 \times 0.3, 1.0 \times 0.02 \times 0.7)]}$
				$[0.4]$
$V \rightarrow$	部分構造に曖昧性が生じた場合は最大スコアとなる部分構造へのバックポインタを保持			
$N \rightarrow$				
$N \rightarrow$				
$P \rightarrow$				
	$win_{[1.0]}$	$[1, 2]$	$[1, 3]$	$[1, 4]$
			$P_{[1.0]}$	$PP_{[0.1]}$
			$[2, 3]$	$[2, 4]$
			$N_{[0.2]}$	$NP_{[0.1]}$
			$[3, 4]$	

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法はチョムスキー標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

				0 $watch$ 1 $girls$ 2 $with$ 3 $glasses$ 4
$V_{[1.0]}$	$VP_{[0.28]}$		$VP_{[0.14]}$ $S_{[0.14 \times 1.0]}$ $[0,4]$	
$[0,1]$	$[0,2]$	$[0,3]$	$X_{[0.04]}$ $NP_{[0.02]}$ $[1,4]$	
$N_{[0.8]}$			$P_{[1.0]}$ $PP_{[0.1]}$ $[2,3]$	
$NP_{[0.4]}$ $[1,2]$		$[1,3]$	$[2,4]$	
			$N_{[0.2]}$ $NP_{[0.1]}$ $[3,4]$	

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキイ標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

$0 \ watch \ 1 \ girls \ 2 \ with \ 3 \ glasses \ 4$			
$V_{[1.0]}$ [0,1]	$VP_{[0.28]}$ [0,2]		$VP_{[0.14]}$ $S_{[0.14]}$ [0,4]
$N_{[0.8]}$ [1,2]	$NP_{[0.4]}$ [1,3]		$X_{[0.04]}$ $NP_{[0.02]}$ [1,4]
		$P_{[1.0]}$ [2,3]	$PP_{[0.1]}$ [2,4]
			$N_{[0.2]}$ $NP_{[0.1]}$ [3,4]

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法はチョムスキー標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

				0 $watch$ 1 $girls$ 2 $with$ 3 $glasses$ 4
$V_{[1.0]}$	$VP_{[0.28]}$		$VP_{[0.14]}$	
$[0,1]$	$[0,2]$	$[0,3]$	$S_{[0.14]}$	
			$[0,4]$	
				$X_{[0.04]}$
				$NP_{[0.02]}$
				$[1,4]$
				$P_{[1.0]}$
				$PP_{[0.1]}$
				$[2,4]$
				$N_{[0.2]}$
				$NP_{[0.1]}$
				$[3,4]$

CKY 表

各セル $[i, j]$ は部分木の根ノードと確率を保持

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法は チョムスキーモードルに変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]}$

$VP \rightarrow V\ NP_{[0.7]}$

$VP \rightarrow V\ X_{[0.3]}$

$X \rightarrow NP\ PP_{[1.0]}$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V \rightarrow watch_{[1.0]}$

$N \rightarrow girls_{[0.8]}$

$N \rightarrow glasses_{[0.2]}$

$P \rightarrow with_{[1.0]}$

時間計算量: $O(T^3)$

CKY 表

各セル $[i, j]$ は部分木の
根ノードと確率を保持

$_0 watch_1 girls_2 with_3 glasses_4$

$V_{[1.0]}$	$VP_{[0.28]}$		$VP_{[0.14]}$
$[0,1]$	$[0,2]$	$[0,3]$	$[0,4]$
$N_{[0.8]}$			$X_{[0.04]}$
$NP_{[0.4]}$			$NP_{[0.02]}$
$[1,2]$	$[1,3]$		$[1,4]$
$P_{[1.0]}$			$PP_{[0.1]}$
$[2,3]$			$[2,4]$
$N_{[0.2]}$			$NP_{[0.1]}$
			$[3,4]$

PCFG のための確率的 CKY アルゴリズム

- CKY アルゴリズムを PCFG を扱えるよう拡張
 - 文法はチョムスキーライントークン標準形に変換して処理
 - CKY 表に部分結果とその確率を格納

$N : S, NP, VP, PP, V, P, N, X$

$\Sigma : watch, girls, with, glasses$

$R : S \rightarrow VP_{[1.0]} \quad | \quad V \rightarrow watch_{[1.0]}$

$VP \rightarrow V\ N D$

$VP \rightarrow V\ N P$

$X \rightarrow N$

$PP \rightarrow P\ NP_{[1.0]}$

$NP \rightarrow NP\ PP_{[0.5]}$

$NP \rightarrow N_{[0.5]}$

$S : S$

$V_{[1.0]}$

$VP_{[0.28]}$

$VP_{[0.14]}$

$S_{[0.14]}$

$X_{[0.04]}$

$NP_{[0.02]}$

$P_{[1.0]}$

$PP_{[0.1]}$

$N_{[0.2]}$

$NP_{[0.1]}$

$[0,1]$

$[0,2]$

$[0,3]$

$[0,4]$

$[1,2]$

$[1,3]$

$[1,4]$

$[2,3]$

$[2,4]$

$[3,4]$

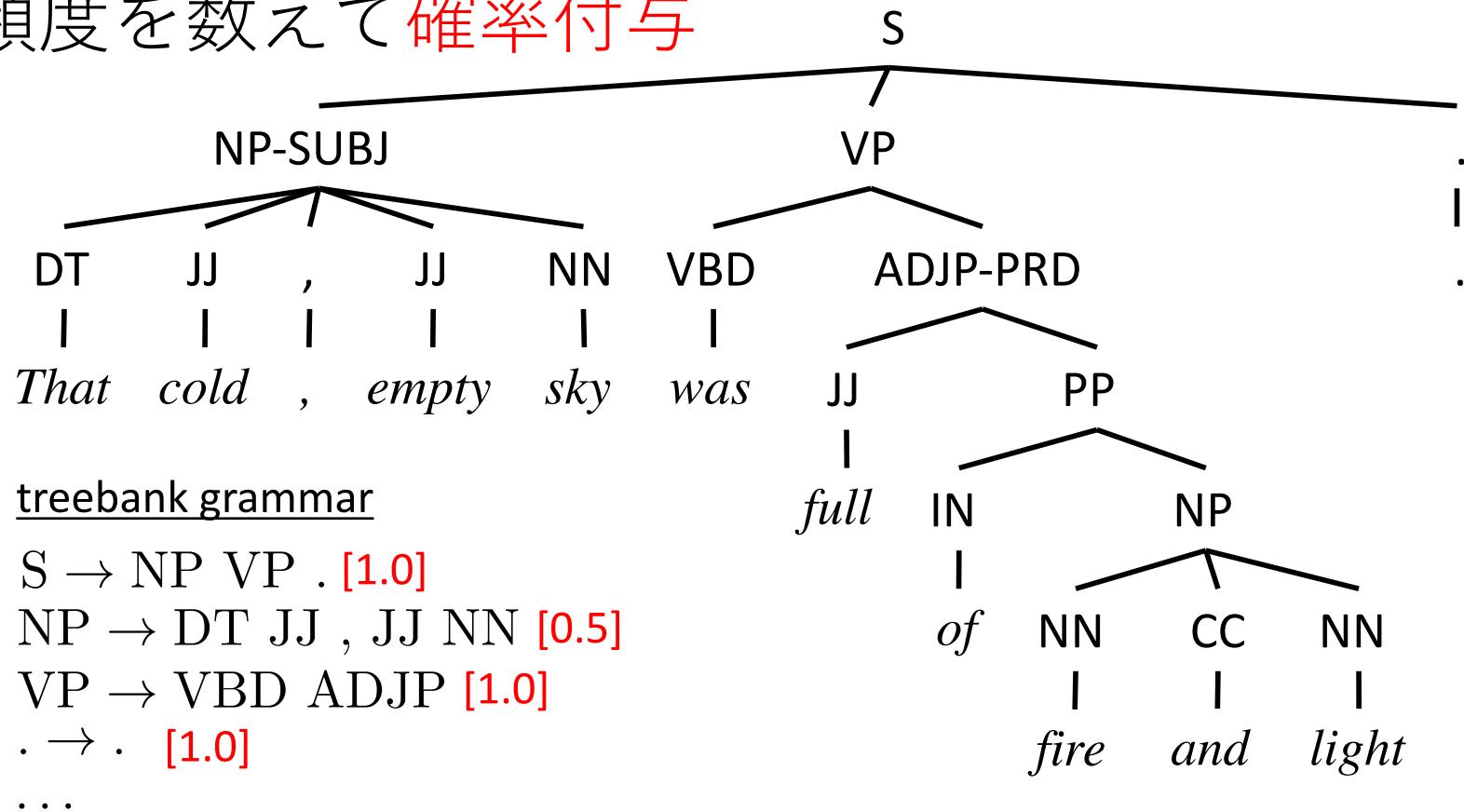
時間計算量: $O(T^3)$

CKY 表

各セル $[i, j]$ は部分木の
根ノードと確率を保持

PCFG の教師あり学習 (獲得)

- PCFG: ツリーバンクを分割することで文法抽出し,
頻度を数えて確率付与



$$P(\beta|A) = \frac{C(A \rightarrow \beta)}{\sum_{\gamma} C(A \rightarrow \gamma)} = \frac{C(A \rightarrow \beta)}{C(A)}$$

単純なツリーバンク PCFG の問題

- 多段の構造的依存関係をモデル化していない



Switchboard コーパス [Francis+ 1999]

	代名詞	-代名詞
主語	91%	9%
目的語	34%	66%

- 語彙的依存関係をモデル化していない

$VP \rightarrow V\ NP\ PP$

or

$VP \rightarrow V\ NP$

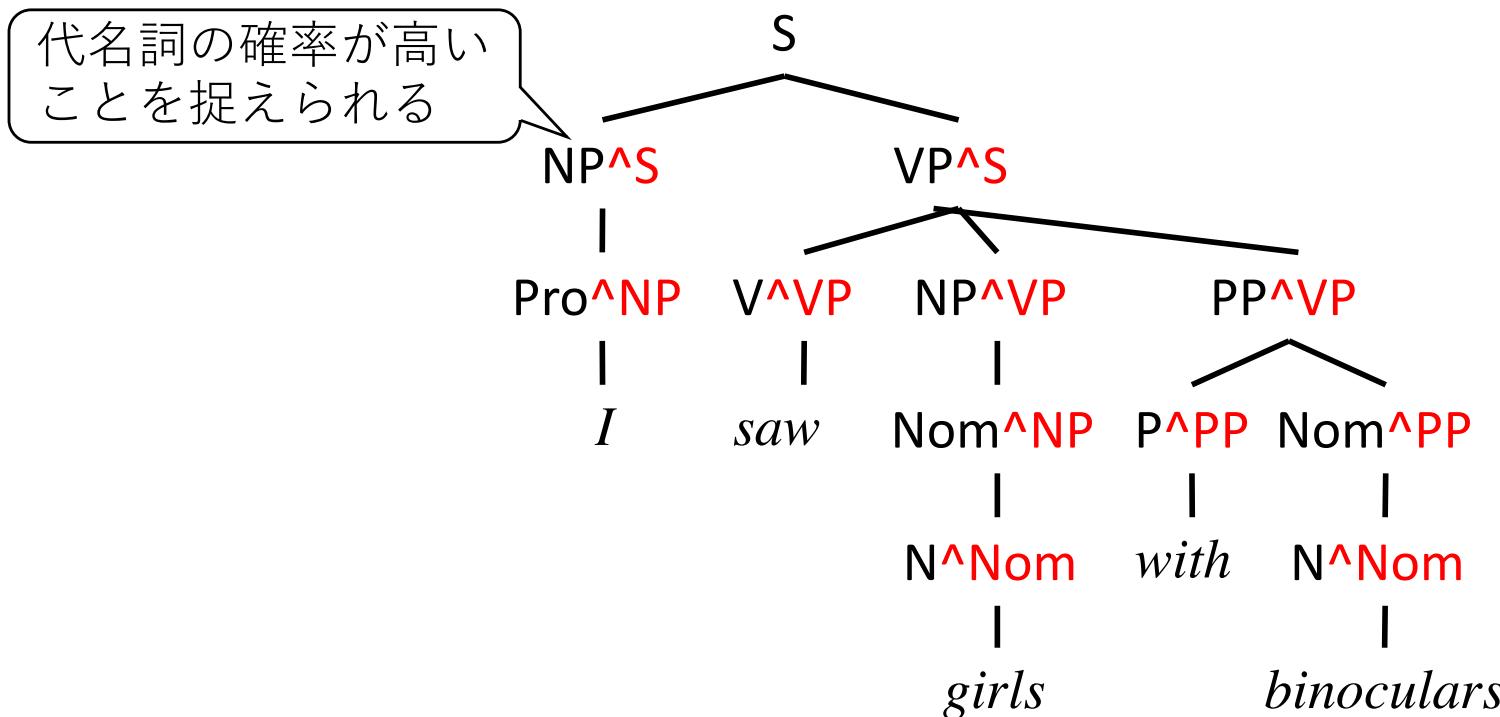
*I saw a girl **with** a telescope*

$NP \rightarrow NP\ PP$

$PP \rightarrow P\ NP$

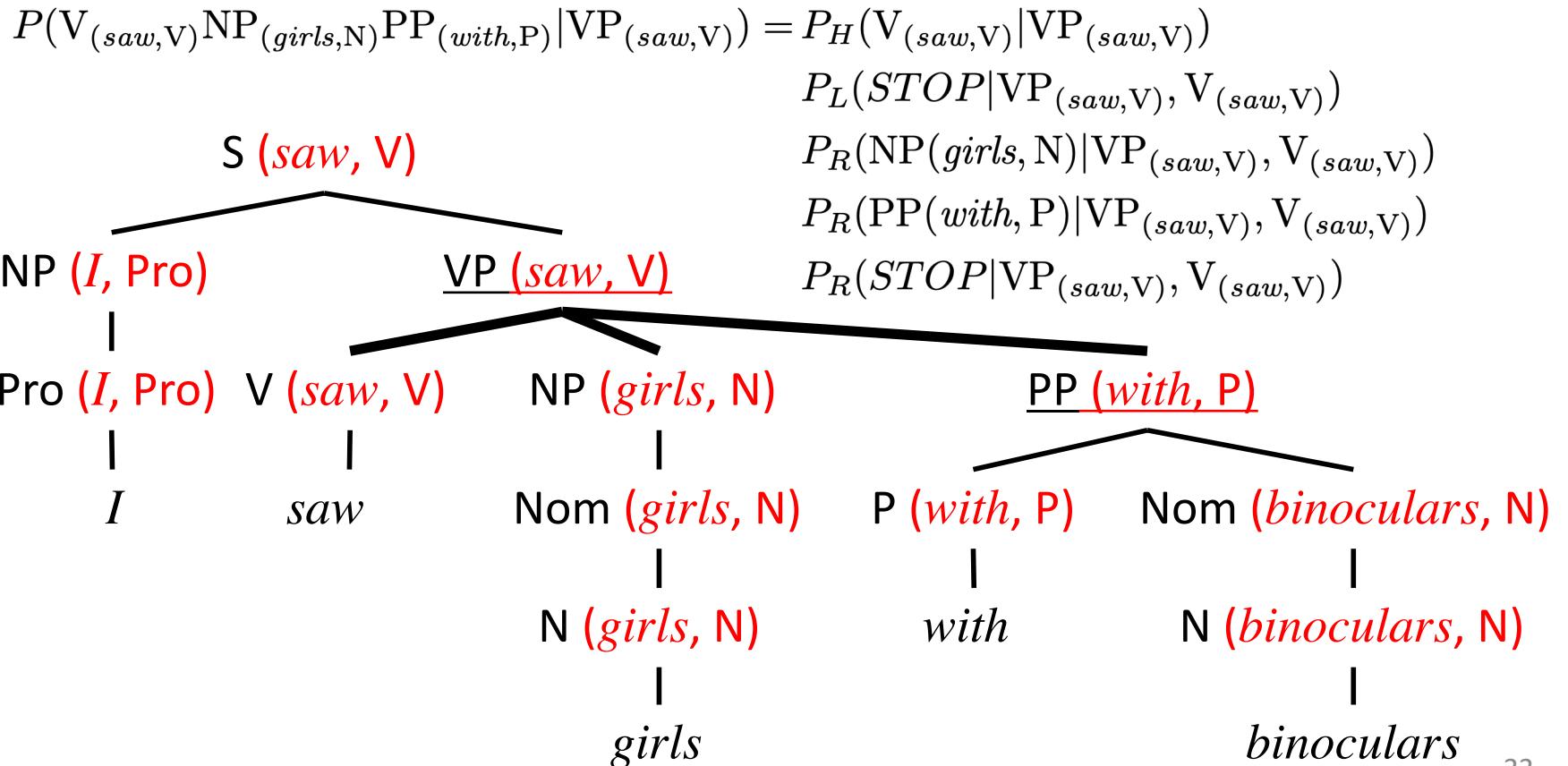
PCFG の問題への対策 (1/2)

- シンボル詳細化により構造的依存関係を捉える
 - parent annotation [Johnson 1998, Klein+ 2003]: 親ノードラベルを付与
 - latent annotation [Matsuzaki+ 2005, Petrov+ 2006]: 自動分割



PCFG の問題への対策 (2/2)

- 語彙化 PCFG [Collins 1999, Charniak 1997]
 - 主辞の単語と非終端記号を親ノードに再帰的に付与
 - ゼロ頻度問題回避のため確率を分解 (+スムージング)



確率的語彙化文法

- 語彙化文法は弱文脈依存性と膨大な語彙項目のため効率的な構文解析を行うことが困難
 - 語彙項目の種類数: 425 (CCGbank) (cf. 45 (Penn Treebank))
 - 時間計算量は $\geq O(n^6)$
- スーパータギング [Bangalore+ 1999]により、語彙項目の重み付けを行った後で構文解析を行う戦略が現実的
 スупータギング: 各単語に語彙項目を割り当てるタスク
 - 周辺化により $P(\text{語彙項目} | w)$ を計算して A* 探索 (省略)
 - スーパータギングにより構文的曖昧性はほとんど解消

発展: PCFG による教師なし構文解析

- 与えられた文の確率が最大になるような PCFG を求める
 - 推定には EM アルゴリズムや変分ベイズ法を使う

統計的句構造解析器の評価

- PARSEVAL [Black+ 1991]: 部分構成素の一致率を計算
正解構文木の完全一致より細やかな評価が可能

$$\text{Labeled Precision} = \frac{\text{システムの出力に含まれる正しい構成素の数}}{\text{システムの出力に含まれる構成素の数}}$$

$$\text{Labeled Recall} = \frac{\text{システムの出力に含まれる正しい構成素の数}}{\text{正解の構文木中の構成素の数}}$$

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2P + R} \quad \text{特に} \quad F_1 = \frac{2PR}{P + R}$$

- Cross-Brackets (CB):
正解の構文木の構成素と包含関係が矛盾する構成素の割合
 $(A (B C))$ vs. $((A, B), C)$

部分構文解析: チャンキング

- チャンキング: 入れ子でない句構造を同定

例) NP チャンキング: 基本名詞句を同定

[The morning flights]_{NP} from [Denver]_{NP} has arrived

- アプローチ: IOBタグ付けに基づく系列ラベリング

- 学習データは既存の句構造ツリーバンクから生成
- 任意の系列ラベリングアルゴリズムを利用可能

The morning flights from Denver has arrived
B_NP I_NP I_NP O I_NP O O

チャンキングの評価

- 個々のラベルの精度ではなく、認識したチャンクの精度(Precision), 再現率(Recall), F_β -measure で行う
 - 正解チャンクとの完全一致より細やかな評価が可能

$$\text{Precision} = \frac{\text{システムの出力に含まれる正解のチャンク数}}{\text{システムの出力に含まれるチャンク数}}$$

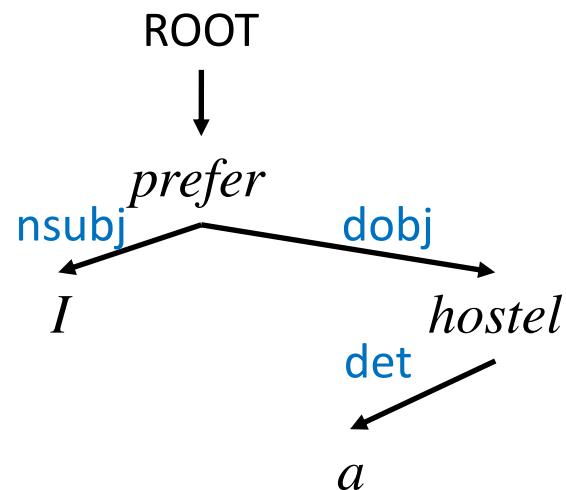
$$\text{Recall} = \frac{\text{システムの出力に含まれる正解のチャンク数}}{\text{正解に含まれるチャンク数}}$$

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2P + R} \quad \text{特に} \quad F_1 = \frac{2PR}{P + R}$$

構文構造を記述する二つのアプローチ

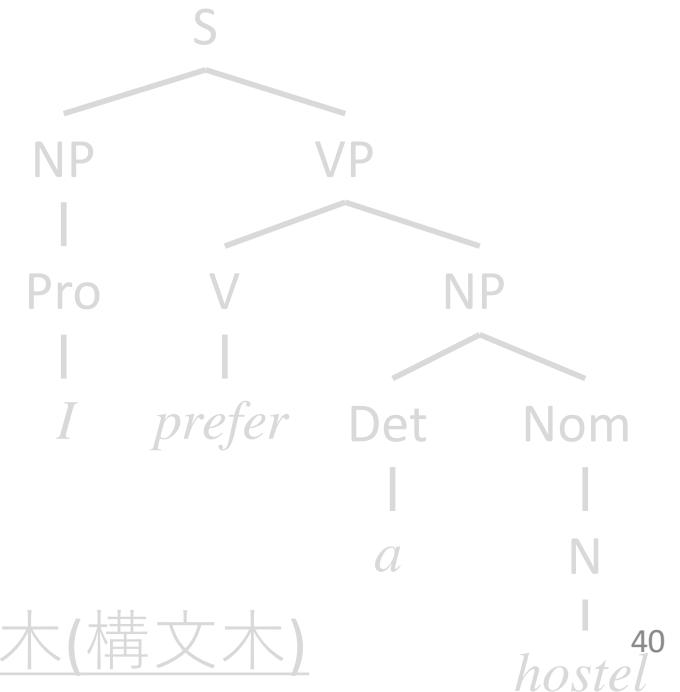
- 依存文法 [T_{esnière} 1959]

- 単語間の統語的依存関係を解析
- 日本語やチェコ語など、語順が自由な言語で発達
- 統計的手法



- 句構造文法 [Chomsky 1956]

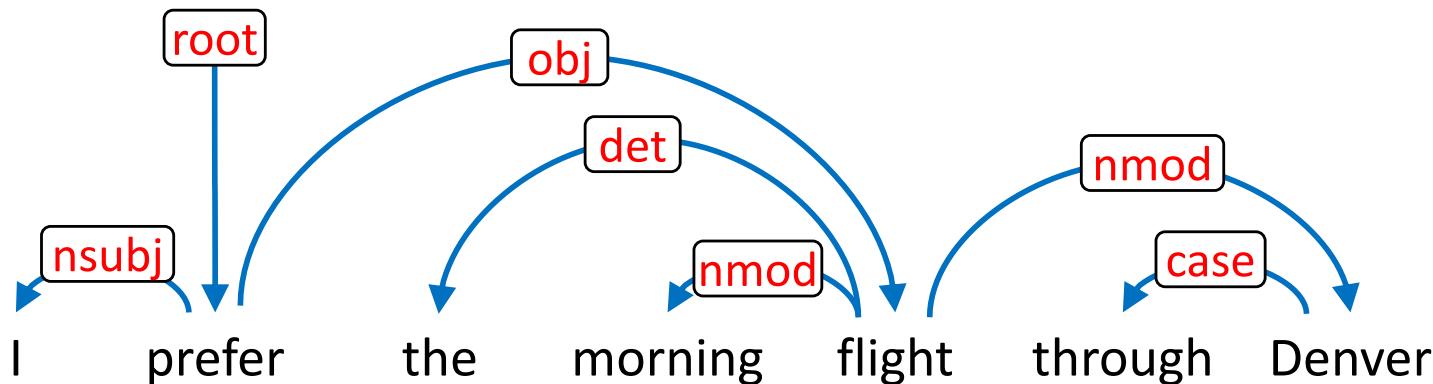
- 単語列(構成素)の階層的包含関係を解析
- 英語など、語順が比較的固定された言語で発達
- 形式文法 + 統計的手法



依存構造解析

与えられた文に対し、(型付き)依存構造を返す

- 従属辞(dependent)と主辞(head)の間の依存関係を弧(arc)
「主辞→従属辞」(と依存関係ラベル)で表現



語順が自由な言語を主な対象として研究が行われていたが
処理効率・言語普遍性・応用での有用性から研究が盛んに

Universal dependenciesで定義された依存関係ラベル

[Nivre+ 2016]

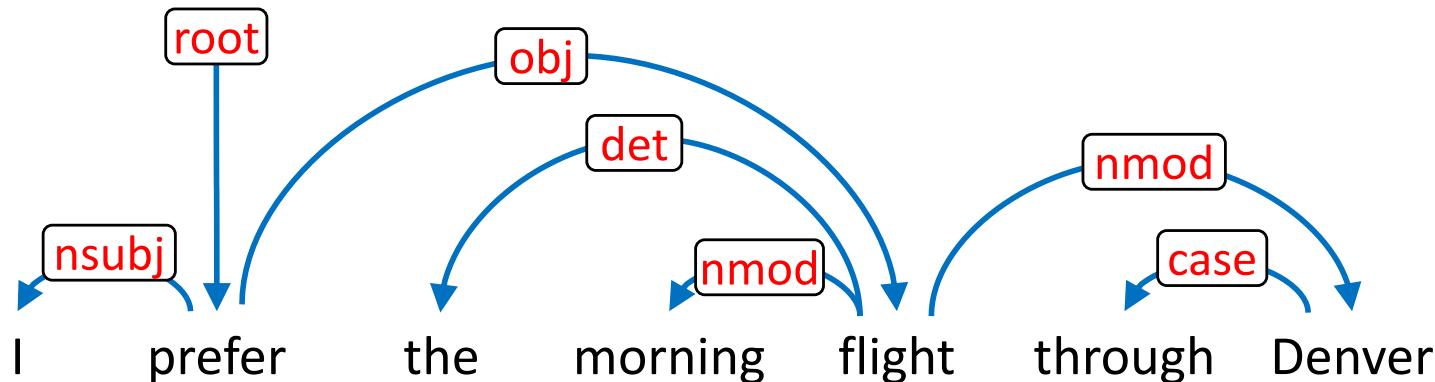
<http://universaldependencies.org/u/dep/index.html> の37関係の中から抜粋

依存関係 例文(主辞と従属辞)

nsubj	United canceled the flights
obj	We booked her the flight to Miami.
iobj	We booked her the flight to Miami.
nmod	We took the morning flight.
amod	Book the cheapest flight.
nummod	Before the storm JetBlue canceled 1000 flights.
appos	United, a unit of UAL, matched the fares
det	The flight was canceled.
conj	We flew to Denver and drove to Steamboat.
cc	We flew to Denver and drove to Steamboat.
case	Book the flight through Houston.

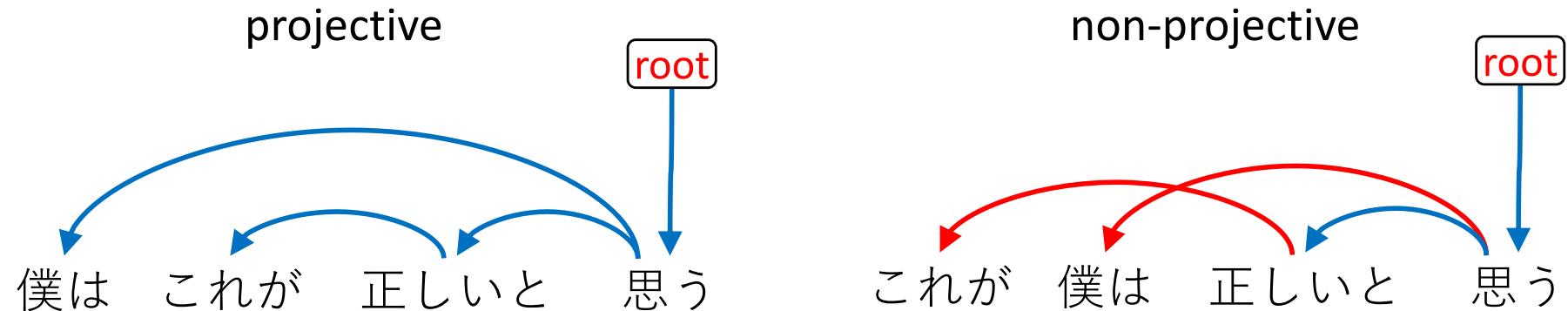
依存文法

- 依存構造木: 単語を節 (node), 単語対 (主辞→従属辞) を弧 (arc) とし, 以下の要件を満たす有向グラフ
 - 従属辞とならないノードがただ一つだけ存在 (root)
 - root 以外の語はただ一つの主辞を持つ (入る弧が一つ)
 - root から各語に至るパスが一意 (非循環)



Projectivity

- Projective な依存構造: 構造中の依存関係が非交差



- 依存構造が projective かどうかにより、適用可能な依存構造解析アルゴリズムが変わる
 - Projective → Graph-based (Eisner), transition-based [Nivre+ 2003]
 - Non-projective → Graph-based (MST) [McDonald+ 2005]

依存構造ツリーバンク

- 依存構造を直接付与したコーパス
 - Prague Dependency Treebank [Bejček et al. 2013]
 - 京都大学テキストコーパス [Kurohashi+ 1994]
- 句構造ツリーバンクから変換する場合も
 - 句構造木中の各分岐に対し、ヒューリスティックで主辞となる子ノードを決定
 - 主辞の子ノードに対し、それ以外の子ノードを従属辞として再帰的に依存関係を復元

依存構造解析アルゴリズム

- Transition-based (greedy)
 - 局所な遷移操作の繰返しにより決定的に構造決定
 - 具体的なアルゴリズム:
arc-standard, arc-eager etc.
- Graph-based
 - 全候補からスコア最大の木を選択
 - 具体的なアルゴリズム:
MST, Eisner

 局所最適

 projective は処理困難

 高速 ($O(n)$)

 複雑な特徴量を使える

 大域最適

 non-projective も処理可

 低速 ($\geq O(n^2)$)

 複雑な特徴量は使えない

依存構造解析アルゴリズム

- Transition-based (greedy)
 - 局所な遷移操作の繰返しにより決定的に構造決定
 - 具体的なアルゴリズム:
arc-standard, arc-eager etc.
- Graph-based
 - 全候補からスコア最大の木を選択(動的計画法)
 - 具体的なアルゴリズム:
MST, Eisner



局所最適



projective は処理困難



高速 ($O(n)$)



複雑な特徴量を使える



大域最適



non-projective も処理可



低速 ($\geq O(n^2)$)



複雑な特徴量は使えない

Transition-based dependency parsing (遷移型依存構造解析)

- 未処理の入力を含む **Buffer** を順に処理し、主辞が未決定の語を **Stack** に保持しつつ依存関係を決定
 1. 初期状態 (**Buffer**, **Stack**, 依存構造) = (入力文, [], [])
 2. 現状態に対し、可能な遷移操作の中から適用する操作を決定し、状態を更新
 3. 2 を終了状態になるまで繰り返す
- 各遷移操作ごとに **Buffer** または **Stack** の要素が必ず一つずつ消費されるので時間計算量は $O(n)$

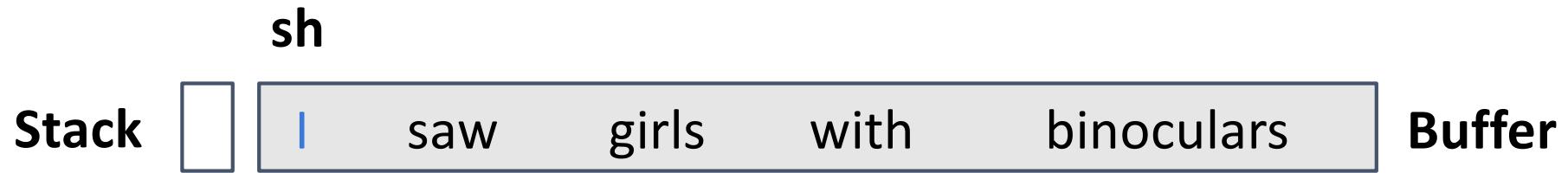
Left-to-right arc-standard strategy

- 未処理の入力を含む **Buffer** を順に処理し、主辞が未決定の語を **Stack** に保持しつつ依存関係を決定



Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入



Stackの要素が2以下のときは shift を決定的に適用

Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入



Stackの要素が2以下のときは shift を決定的に適用

Left-to-right arc-standard strategy

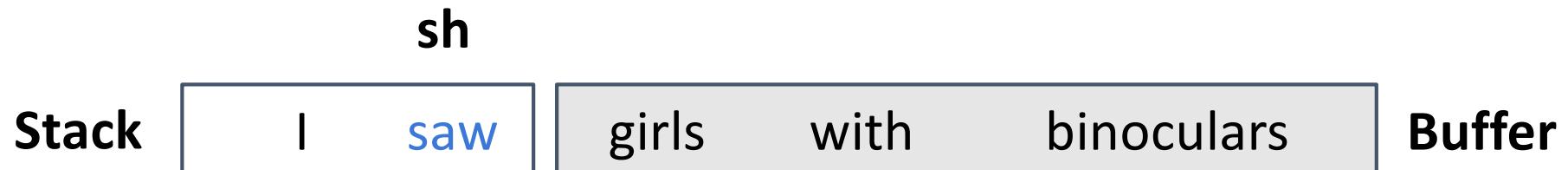
- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入



Stackの要素が2以下のときは shift を決定的に適用

Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入



Stackの要素が2以下のときは shift を決定的に適用

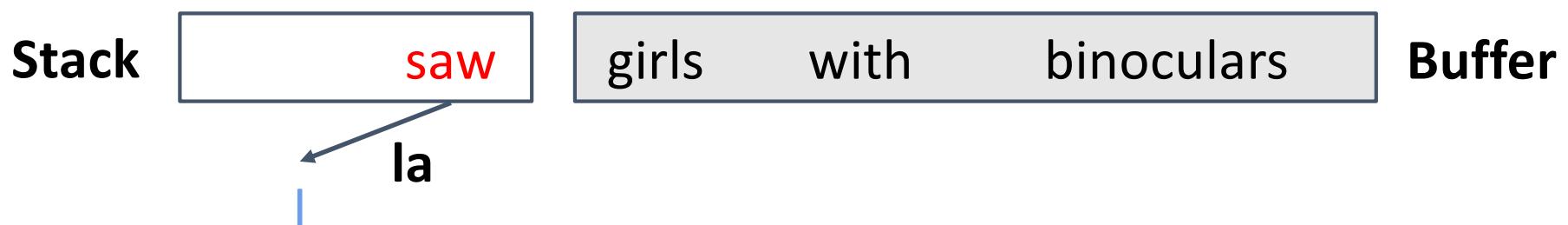
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



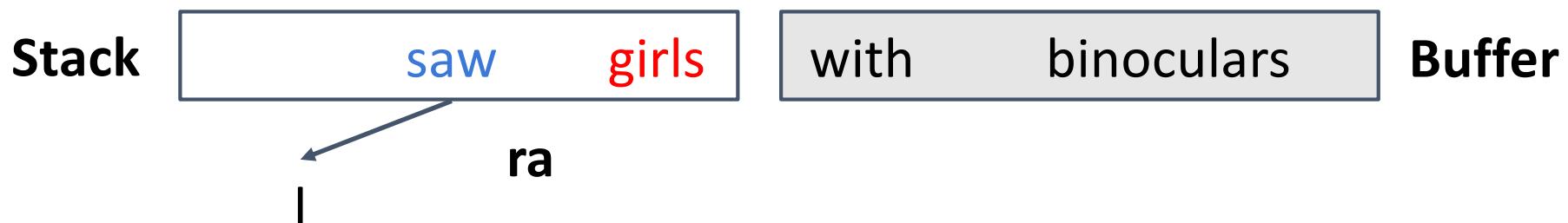
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



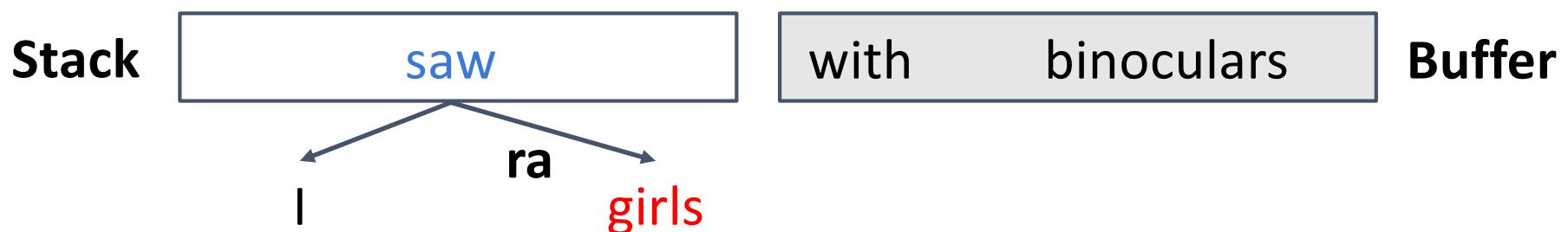
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



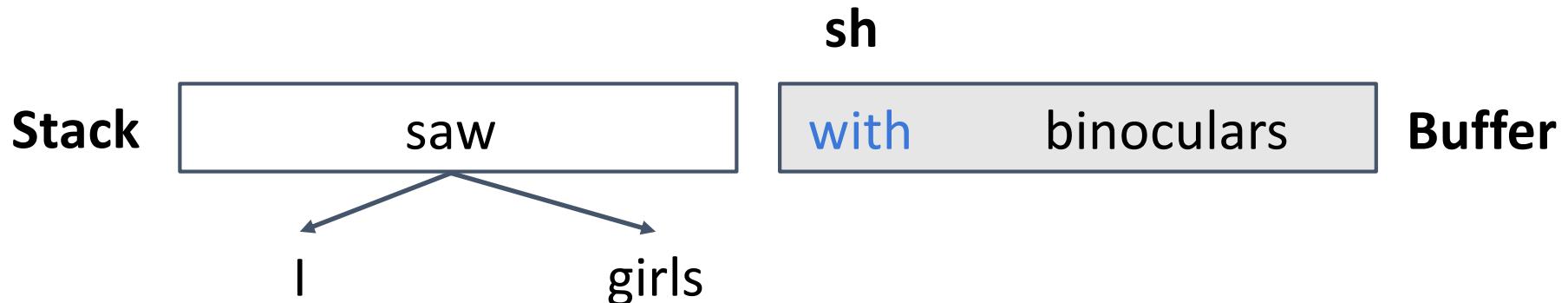
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



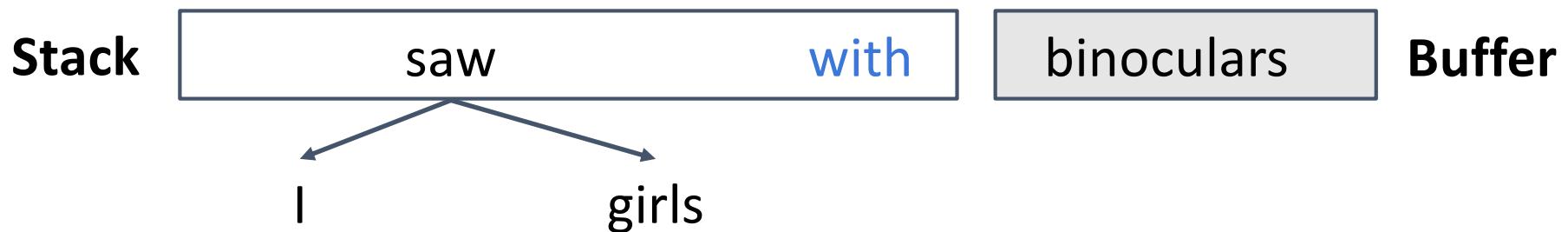
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



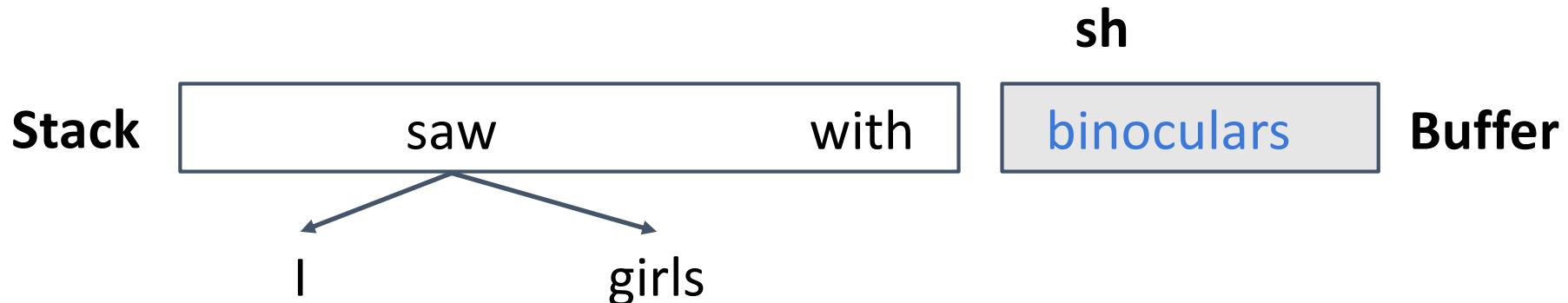
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



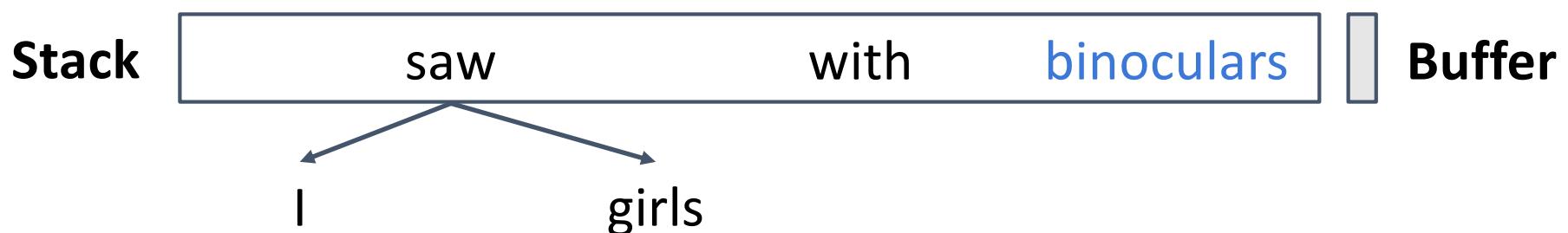
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



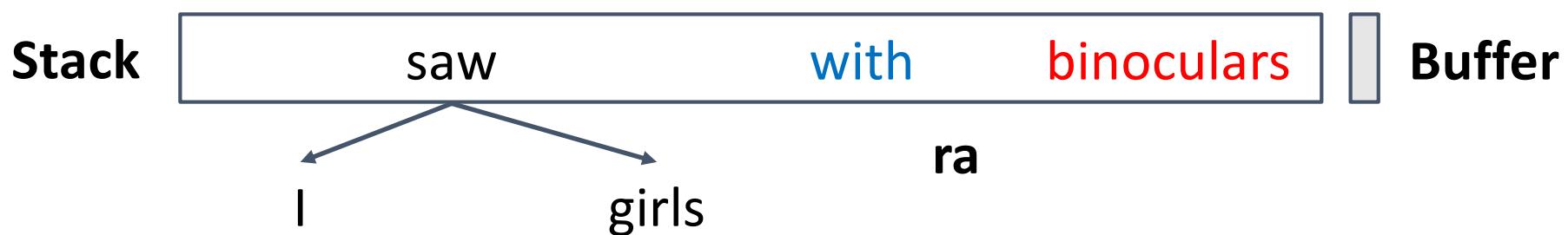
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



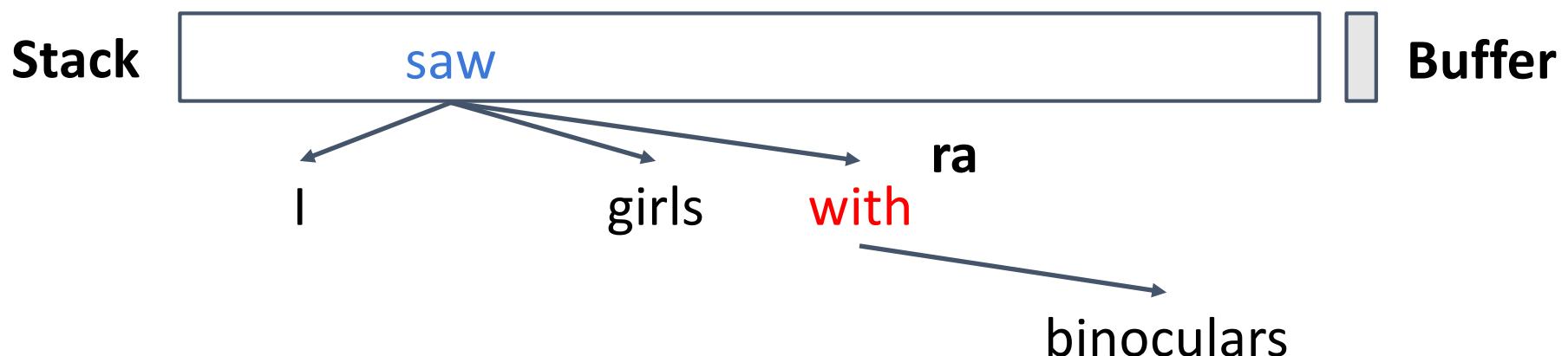
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



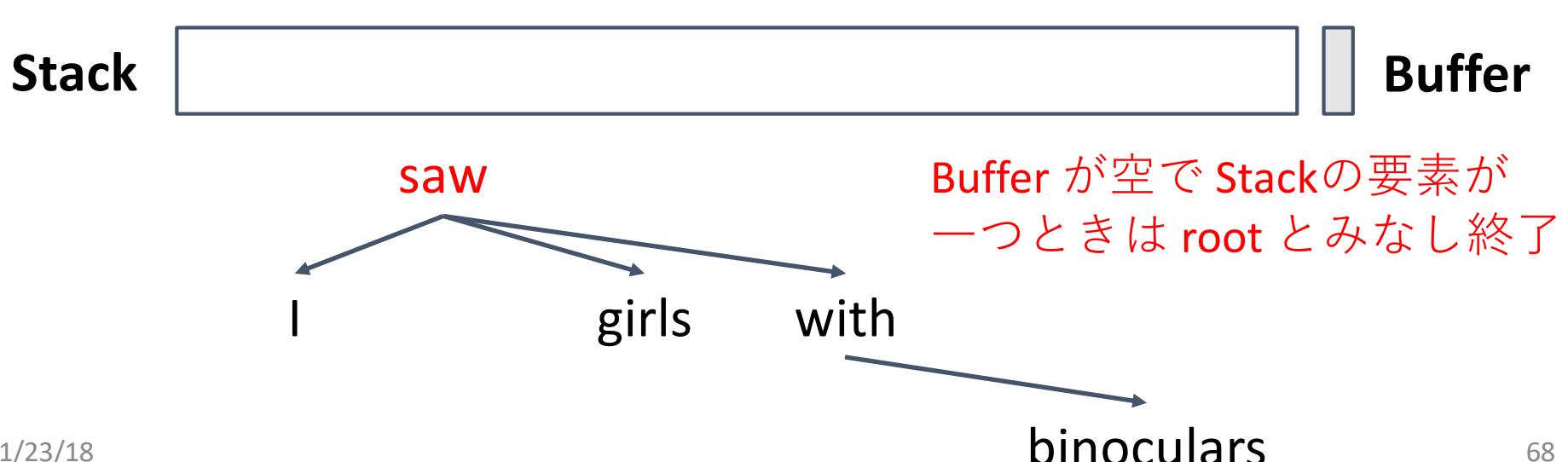
Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ 依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



Left-to-right arc-standard strategy

- 未処理の入力を含む Buffer を順に処理し、主辞が未決定の語を Stack に保持しつつ依存関係を決定
 - shift: Buffer (の先頭) を Stack に挿入
 - left-arc: Stack (top) から Stack (next) への弧を追加
 - right-arc: Stack (next) から Stack (top) への弧を追加



遷移型依存構造解析器の学習 (1/2): 正解データの生成

- 各状態に対して適切な遷移を選ぶ問題を多クラス分類問題として定式化 (部分問題への分割)
 - 依存構造木 = 遷移操作列
- 正解の依存構造木に対して初期状態から正解の木が outputされるように正解の遷移操作(oracle)を決定
 - Stack (top) から Stack (next) への弧が存在 → left-arc
 - Stack (next) から Stack (top) への弧が存在し,
かつ Stack (top) への弧が全て出力されている → right-arc
 - 上記以外 → shift

Stack から取り出された単語に新しい弧を追加することができないため

遷移型依存構造解析器の学習 (2/2): 特徴量の設計と重みの学習

- 特徴量: 以下の基本特徴の組み合わせ
 - Stack / Buffer 中の単語の語形/lemma/品詞/単語埋め込み
 - 出力済みの依存関係
 - 主辞と従属辞の距離



- 分類器: 多クラスロジスティック回帰, Perceptron etc.

ビームサーチに基づく非決定的依存構造解析

- アイデア: 遷移型依存構造解析を遷移操作スコアの和が最大の遷移操作列を選ぶ問題とする
 - 実は大域最適化可能 (多項式オーダ) [Huang+ 2006, Kuhlmann+ 2011]
- 実装: agenda(サイズ β)に解析状態と遷移操作スコアの和を複数保持し, 非決定的に解析
 1. 初期状態 ((Buffer, Stack, 依存構造) = [入力文, [], []]) を agenda に追加
 2. agenda 中の各状態に対して各遷移のスコアを計算
 3. 2 で生成したスコア上位 β の状態で agenda を更新し, agenda に非終了状態が含まれる限り 2 を繰り返す
 4. agenda 中のスコア最大の終了状態を出力

依存構造解析アルゴリズム

- Transition-based (greedy)
 - 局所な遷移操作の繰返しにより決定的に構造決定
 - 具体的なアルゴリズム:
arc-standard, arc-eager etc.
- Graph-based
 - 全候補からスコア最大の木を選択(動的計画法)
 - 具体的なアルゴリズム:
MST, Eisner



局所最適



projective は処理困難



高速 ($O(n)$)



複雑な特徴量を使える



大域最適



non-projective も処理可



低速 ($\geq O(n^2)$)



複雑な特徴量は使えない

Graph-based dependency parsing (グラフに基づく依存構造解析)

- 全候補からスコア最大となる依存構造木を選択

$$\hat{T}(S) = \operatorname{argmax}_{T \in \mathcal{G}_S} score(S, T)$$

- Edge-factored モデル: 依存関係間の独立性を仮定し各依存関係のスコアの和で全体のスコアを計算

$$\hat{T}(S) = \operatorname{argmax}_{T \in \mathcal{G}_S} \sum_{e \in T} score(S, e)$$

- Non-projective な依存構造を扱う場合、最小全域有向木を求めるアルゴリズム [Chu&Liu 1965, Edmonds 1967] で $O(n^2)$
- Projective な依存構造を扱う場合は $O(n^3)$ [Eisner 1996]

エッジスコアの学習

- 各エッジを素性関数によりモデル化

$$\begin{aligned}\hat{T}(S) &= \operatorname{argmax}_{T \in \mathcal{G}_S} \sum_{e \in T} \text{score}(S, e) \\ &= \operatorname{argmax}_{T \in \mathcal{G}_S} \sum_{e \in T} \mathbf{w}^T \phi(S, e)\end{aligned}$$

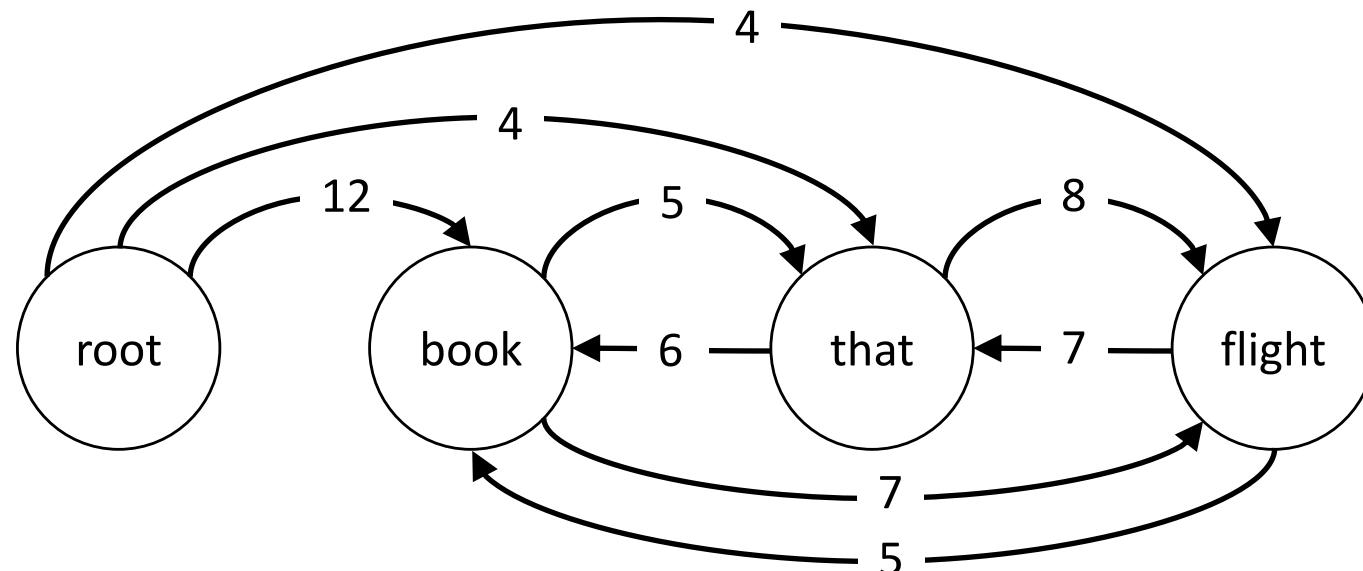
- 特徴量: 遷移型依存構造解析の分類器と同じ
 - 主辞, 従属辞, 周辺語の語形/lemma/品詞/単語埋め込み
 - 主辞と従属辞の距離
- 分類器: 構造化 Perceptron (or PA)など
 - 正解と学習中モデルの出力が一致しないとき正解エッジのスコアを上げ不正解エッジのスコアを下げるよう更新

最小全域有向木の探索による依存構造解析: Chu-Liu/Edmondsアルゴリズム [Chu & Liu 1965; Edmonds 1967]

- 最小全域有向木の探索 = 依存構造解析 [Macdonald+ 2005]

重み付き有向グラフで全ノードを繋ぐ重み最小の木

1. 各ノード v に入る弧 (u,v) 中で、スコア最大の有向弧 $((h,v); h:\text{主辞})$ を選択し、そのスコアを (u,v) から引く
2. 選択された弧が循環する場合、循環するノードを一つに縮退して1を繰り返す
3. 循環が無くなれば、縮退したノードを展開する

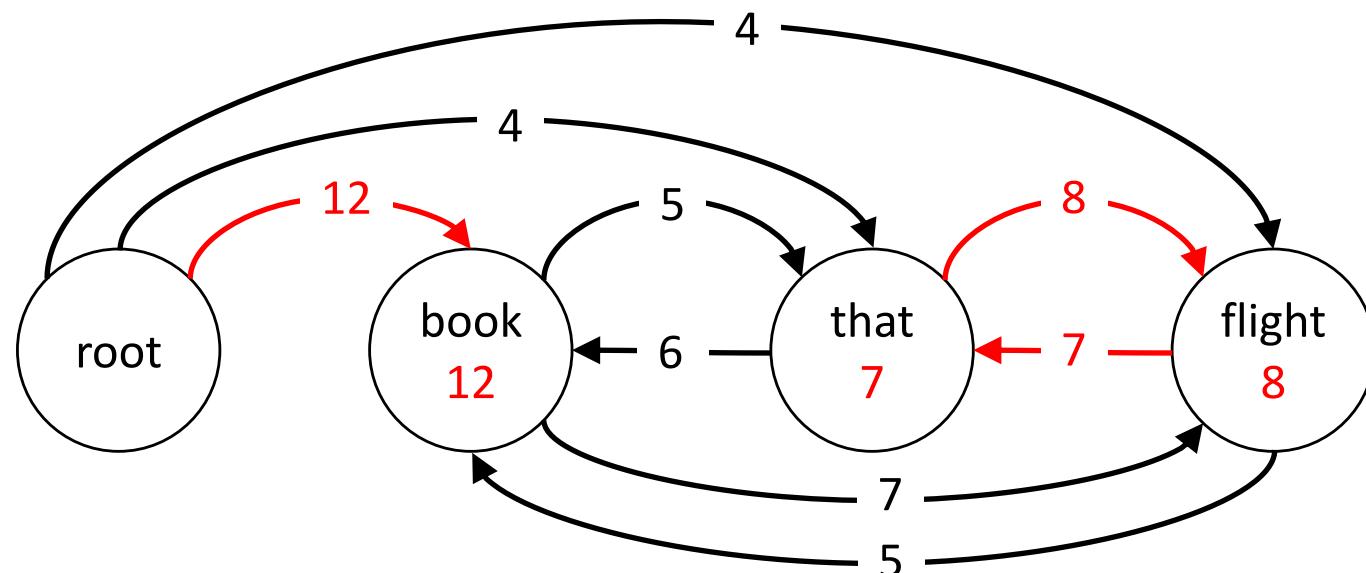


最小全域有向木の探索による依存構造解析: Chu-Liu/Edmondsアルゴリズム [Chu & Liu 1965; Edmonds 1967]

- 最小全域有向木の探索 = 依存構造解析 [Macdonald+ 2005]

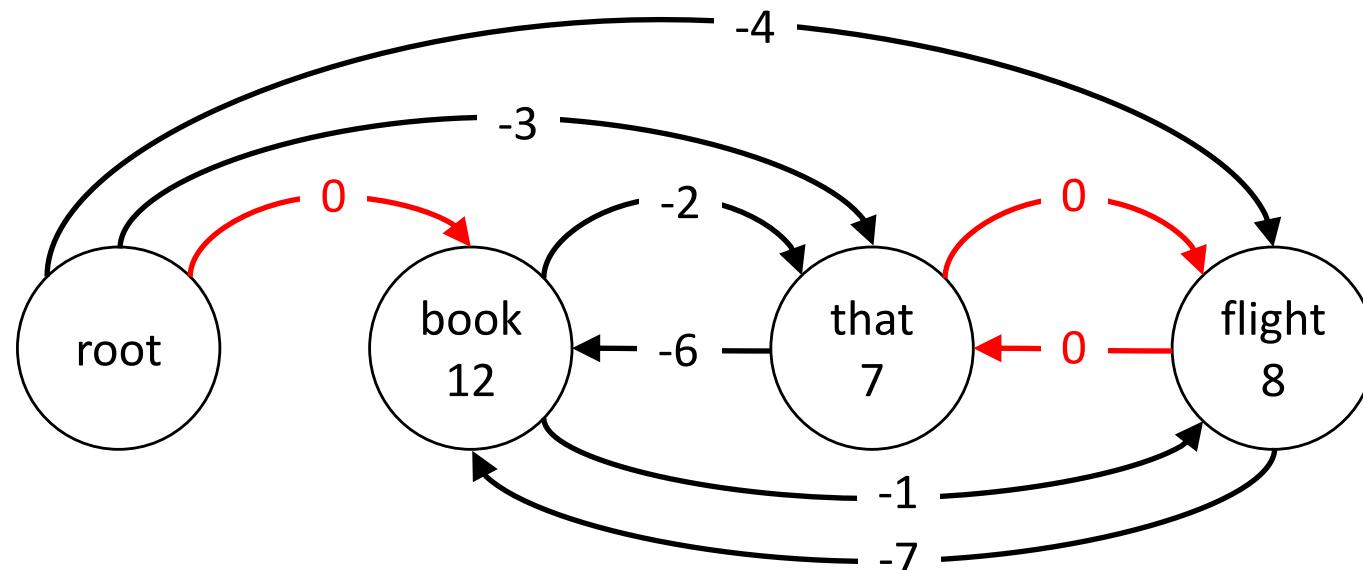
重み付き有向グラフで全ノードを繋ぐ重み最小の木

1. 各ノード v に入る弧 (u,v) 中で、**スコア最大の有向弧 $((h,v); h:\text{主辞})$ を選択**し、そのスコアを (u,v) から引く
2. 選択された弧が循環する場合、循環するノードを一つに縮退して1を繰り返す
3. 循環が無くなれば、縮退したノードを展開する



最小全域有向木の探索による依存構造解析: Chu-Liu/Edmondsアルゴリズム [Chu & Liu 1965; Edmonds 1967]

- 最小全域有向木の探索 = 依存構造解析 [Macdonald+ 2005]
重み付き有向グラフで全ノードを繋ぐ重み最小の木
 1. 各ノード v に入る弧 (u,v) 中で, **スコア最大の有向弧 $((h,v); h:\text{主辞})$ を選択し, そのスコアを (u,v) から引く**
 2. 選択された弧が循環する場合, 循環するノードを一つに縮退して1を繰り返す
 3. 循環が無くなれば, 縮退したノードを展開する

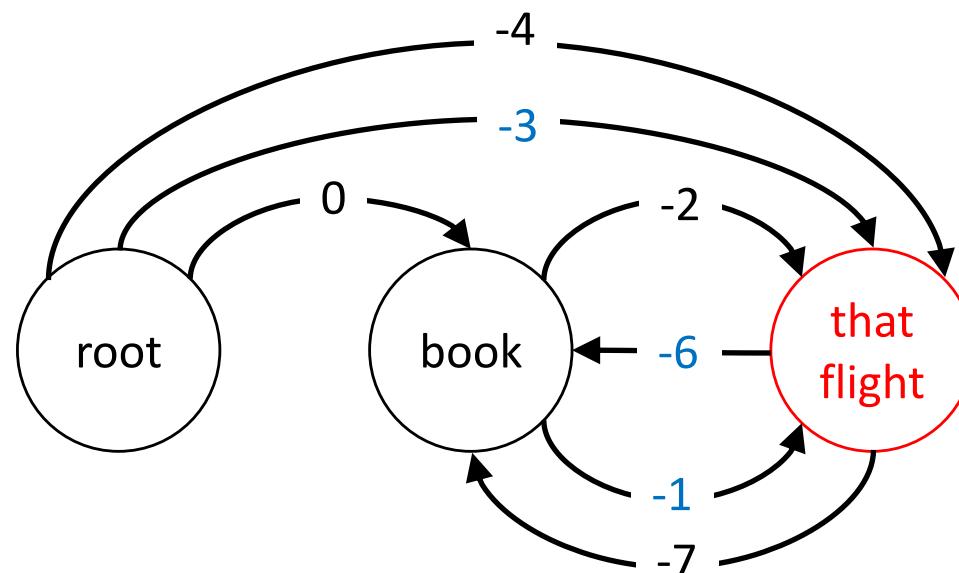


最小全域有向木の探索による依存構造解析: Chu-Liu/Edmondsアルゴリズム [Chu & Liu 1965; Edmonds 1967]

- 最小全域有向木の探索 = 依存構造解析 [Macdonald+ 2005]

重み付き有向グラフで全ノードを繋ぐ重み最小の木

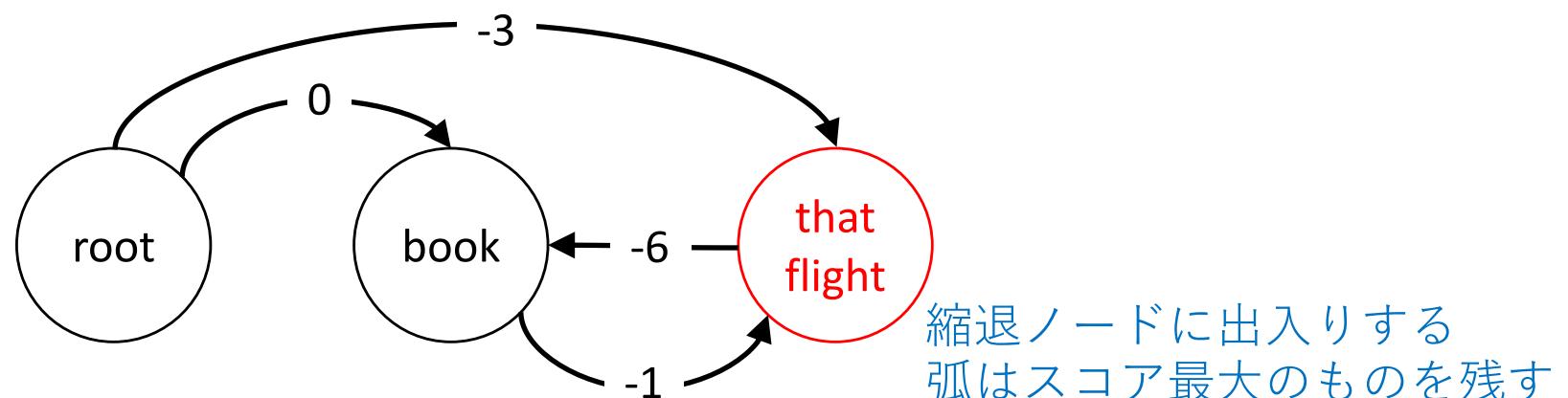
1. 各ノード v に入る弧 (u,v) 中で、スコア最大の有向弧 $((h,v); h:\text{主辞})$ を選択し、そのスコアを (u,v) から引く
2. 選択された弧が循環する場合、**循環するノードを一つに縮退**して1を繰り返す
3. 循環が無くなれば、縮退したノードを展開する



縮退ノードに出入りする
弧はスコア最大のものを残す

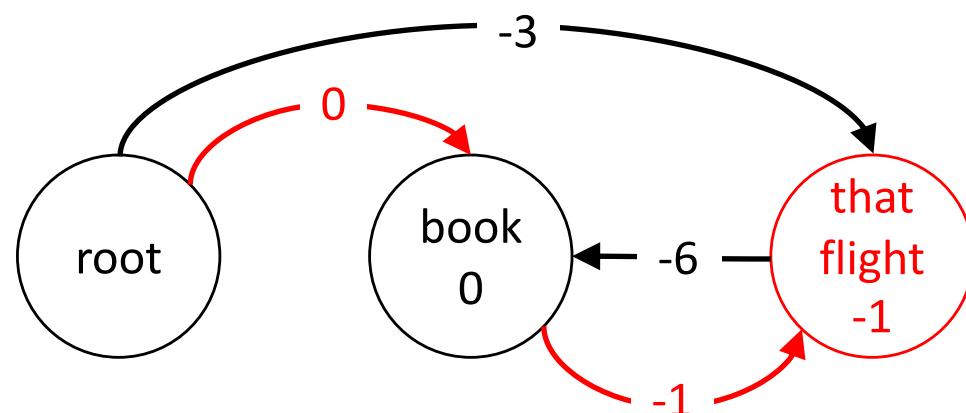
最小全域有向木の探索による依存構造解析: Chu-Liu/Edmondsアルゴリズム [Chu & Liu 1965; Edmonds 1967]

- 最小全域有向木の探索 = 依存構造解析 [Macdonald+ 2005]
重み付き有向グラフで全ノードを繋ぐ重み最小の木
 1. 各ノード v に入る弧 (u,v) 中で、スコア最大の有向弧 $((h,v); h:\text{主辞})$ を選択し、そのスコアを (u,v) から引く
 2. 選択された弧が循環する場合、循環するノードを一つ
に縮退して1を繰り返す
 3. 循環が無くなれば、縮退したノードを展開する



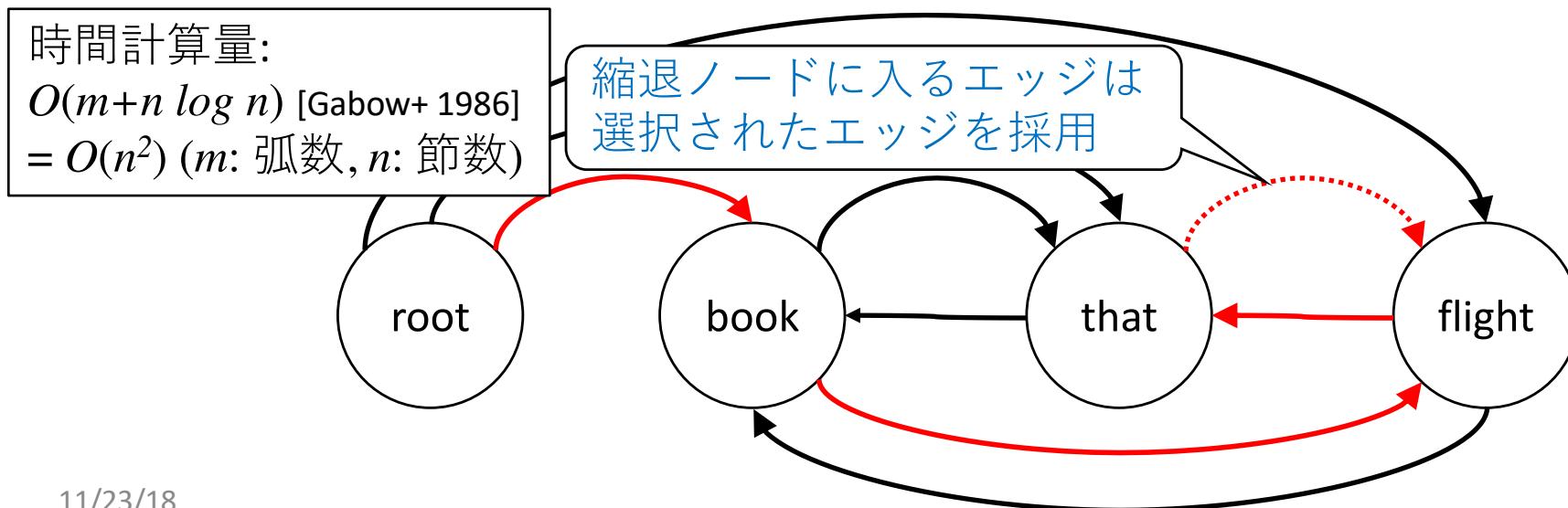
最小全域有向木の探索による依存構造解析: Chu-Liu/Edmondsアルゴリズム [Chu & Liu 1965; Edmonds 1967]

- 最小全域有向木の探索 = 依存構造解析 [Macdonald+ 2005]
重み付き有向グラフで全ノードを繋ぐ重み最小の木
 1. 各ノード v に入る弧 (u,v) 中で、スコア最大の有向弧 $((h,v); h:\text{主辞})$ を選択し、そのスコアを (u,v) から引く
 2. 選択された弧が循環する場合、循環するノードを一つに縮退して**1を繰り返す**
 3. 循環が無くなれば、縮退したノードを展開する



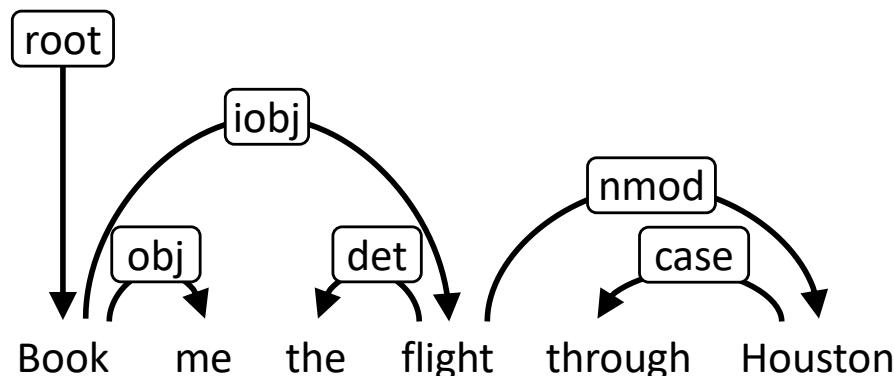
最小全域有向木の探索による依存構造解析: Chu-Liu/Edmondsアルゴリズム [Chu & Liu 1965; Edmonds 1967]

- 最小全域有向木の探索 = 依存構造解析 [Macdonald+ 2005]
重み付き有向グラフで全ノードを繋ぐ重み最小の木
 1. 各ノード v に入る弧 (u,v) 中で、スコア最大の有向弧 $((h,v); h: \text{主辞})$ を選択し、そのスコアを (u,v) から引く
 2. 選択された弧が循環する場合、循環するノードを一つに縮退して1を繰り返す
 3. 循環が無くなれば、縮退したノードを展開する

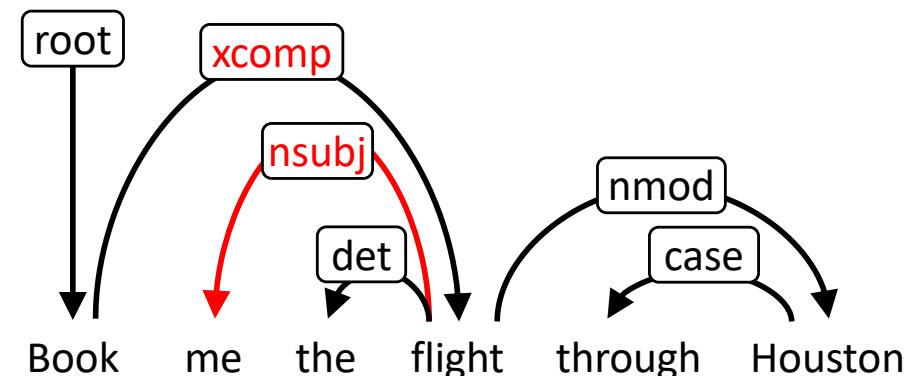


依存構造解析の評価

- 正解を参照し依存関係単位で評価
 - UAS (Unlabeled Attachment Score): 依存関係の一致度
 - LAS (Labeled Attachment Score): 型付き依存関係の一致度
 - LS (Label accuracy Score): 単語単位のラベル正解率
- 依存関係のラベルごとの精度/再現率



Reference



System

LAS = 4/6
UAS = 5/6
LS = 4/6

人間はどのように文の構造を解析するのか？

- 与えられた文に対し、人の読了時間(注視時間)は
 - 言語モデルによる単語の予測確率と相関 [Scott+ 2003]
 - 単語直前の構文構造による単語の予測確率と相関 [Hale 2001]
- 袋小路文 (garden-path 文) では読了時間が増加する
 - 人が文を先頭から逐次的に構文解析している証拠となる

The horse raced past the barn fell.

その馬は納屋を通過して走った ... fell ?
納屋を通過して走らされた馬が転んだ

縮約関係詞節の確率が低いことを示唆

本日のまとめ

- 統計的句構造解析
 - PCFG (確率的文脈自由文法)
 - 確率的 CKY アルゴリズムによる曖昧性解消
 - 高精度のための工夫: シンボル詳細化, 語彙化
 - 確率的語彙化文法
 - スーパータギングによる語彙項目の絞り込み
- 依存構造解析
 - Transition-based アルゴリズム: arc-standard [Nivre 2003]
 - Graph-based アルゴリズム: MST [McDonald 2005]
- 人の構文解析