人工知能特論II　第15回

二宮　崇

今日の講義の予定

- CFGの条件付確率場
- 係り受け解析 (MSTパージング)
- 決定的構文解析

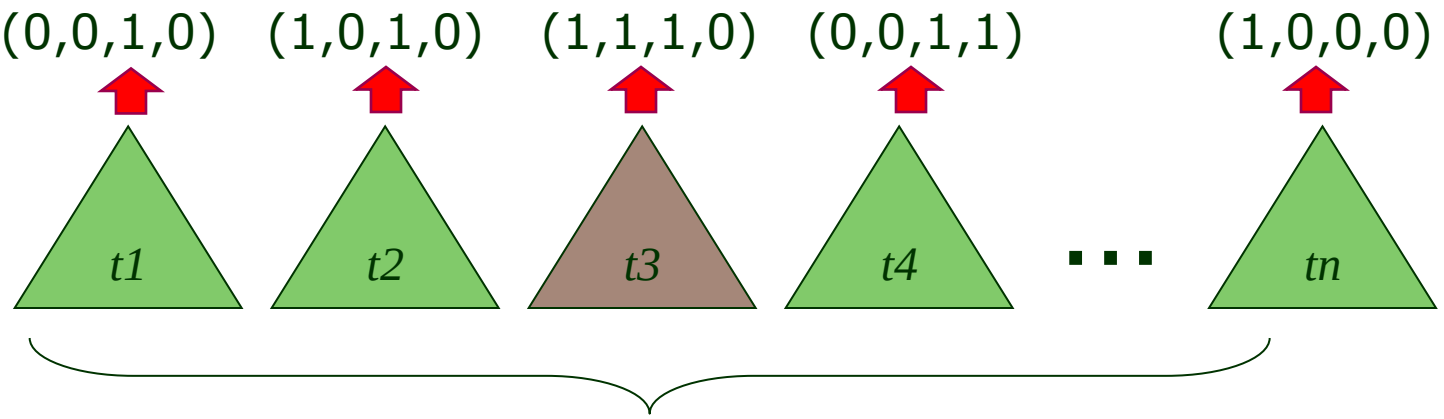
Conditional Random Fields for CFGs

CFGの条件付確率場 (CRF)

識別モデル

$p(t \mid s)$ S＝“A blue eye girl with white hair and skin walked”

素性ベクトル
(特徴ベクトル)

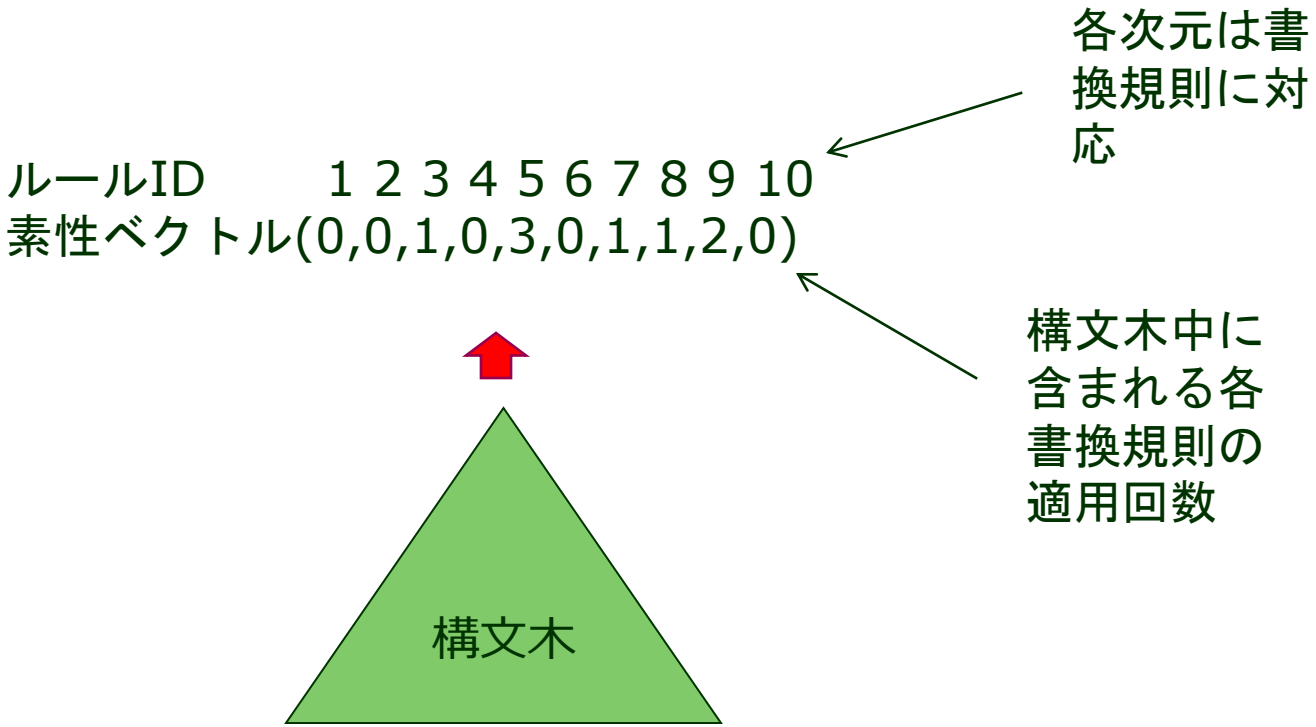


文法Gによりsから導出出来る全ての構文木集合

$p(t3|s)$ はt1,t2,t3,..,tnからt3を選択する確率

CFGの識別モデルの例

- 構文木生成に用いられた各書換規則の適用回数

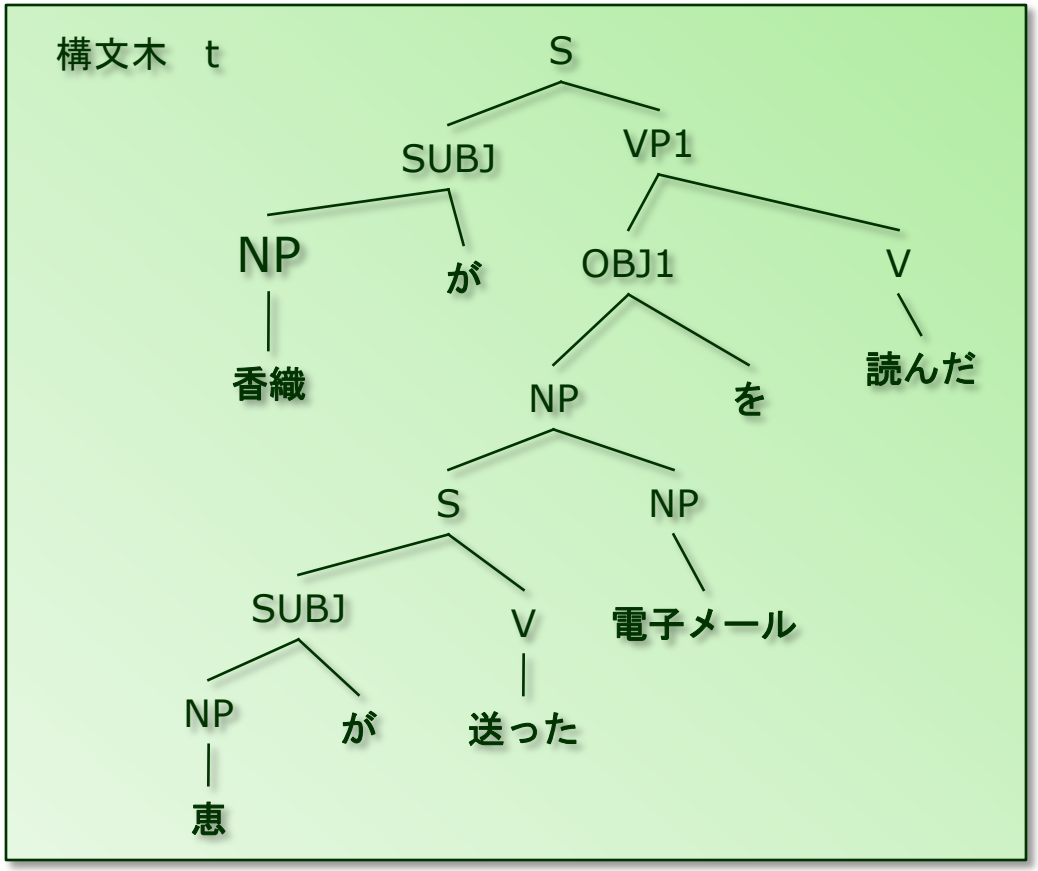


構文木の素性ベクトル

簡単なCFGの例	ID
S → SUBJ　VP1	1
S → SUBJ　V	2
SUBJ → NP　が	3
VP1 → OBJ1　V	4
OBJ1 → NP　を	5
NP → S　NP	6
V → 送った	7
V → 読んだ	8
NP → 香織	9
NP → 恵	10
NP → 電子メール	11
NP → プレゼント	12
NP → 香織　NP1	13
NP → 恵　NP1	14
NP1 → と　NP	15

ID 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

素性ベクトル(1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0)



素性森 (Feature Forest)
畳み込み構文森のためのCRF (Packed Parse CRF)

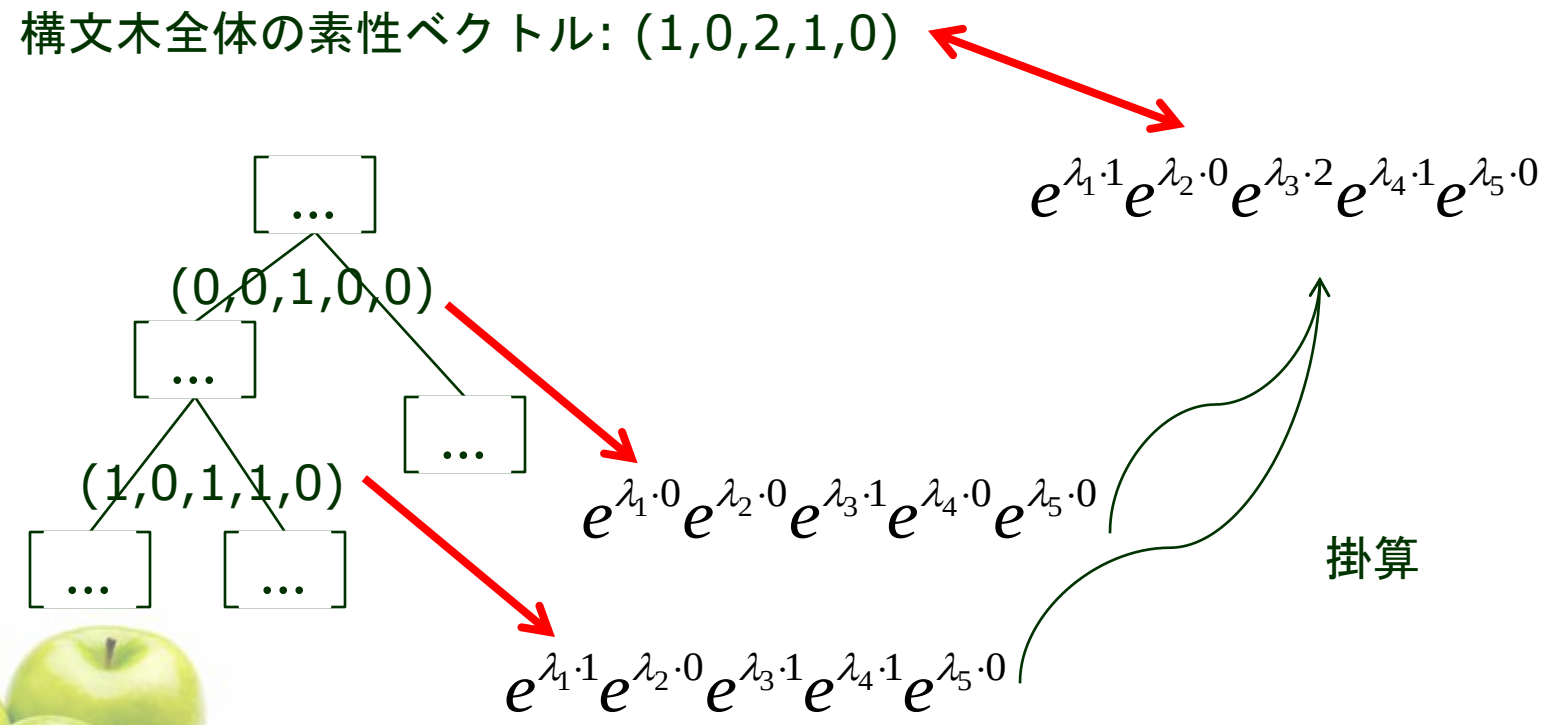
- 素性関数の期待値の計算: 「ある文xに対する全ての構文木集合Y(x)に対する確率」を計算しないといけない

$$\frac{\partial L(\lambda)}{\partial \lambda_j} = \sum_{i=1}^{N} f_j(x_i, y_i) - \sum_{i=1}^{N} \sum_{y \in Y(x_i)} p(y \mid x_i; \lambda) f_j(x_i, y)$$

- 畳み込まれたデータ構造を展開することなく素性関数の期待値を計算
  - 内側外側アルゴリズム (構文木集合)
  - 前向き後向きアルゴリズム (系列ラベリング)

素性森

- 各ブランチのスコアの積＝全体のスコア



素性森

- 構文木の確率

$$p(y \mid x;\lambda) = \frac{1}{Z(x,\lambda)} \exp\left(\sum_j \lambda_j f_j(x,y)\right)$$

$Z(x,\lambda)$：構文木集合全体の確率の和（＝文全体に対する内側確率）

$$\exp\left(\sum_j \lambda_j f_j(x,y)\right) = \prod_j e^{\lambda_j f_j(x,y)} = \prod_c \prod_j e^{\lambda_j f_j(c)}$$

c：構文木の各ブランチ

PCFGの書換規則の確率に対応

- 内側外側アルゴリズムの適用
  - 書換規則の適用回数⇒素性値(素性の発火回数)
  - 書換規則の確率 θ_r ⇒ブランチのスコア $\prod_j e^{\lambda_j f_j(c)}$

まとめ

	品詞解析	構文解析
データ構造	曖昧性のある畳み込まれた列	曖昧性のある畳み込まれた木構造
生成モデル	HMM	PCFG
生成モデルの教師無し学習	最尤法(EMアルゴリズム)+前向き後ろ向きアルゴリズム	最尤法(EMアルゴリズム)+内側外側アルゴリズム
生成モデルの教師付学習	正解データの頻度から計算	正解データの頻度から計算
確率的識別モデル	Linear-Chain CRF	Feature Forest (Packed-Parse CRF)
識別モデルの教師付学習(教師付き学習)	最尤法(主に勾配法)+前向き後ろ向きアルゴリズム	最尤法(主に勾配法)+内側外側アルゴリズム

Deterministic Parsing

決定性構文解析

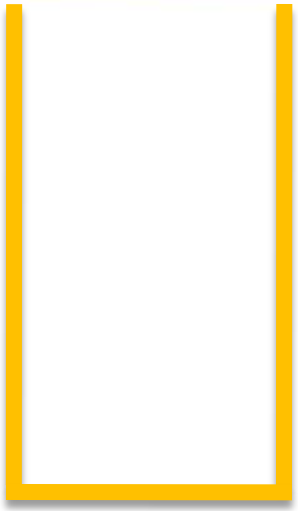
動的計画法の問題点

- 大域素性が使えない
  - 部分構文木を作る順序に依存しない→左や右の外側の部分構文木の情報を使えない
  - 畳込みのために子ノード以下の内側の部分構文木の情報が使えない
- データ表現に制限
  - 意味構造等を構文木ノードに付随させると意味構造も分解し、構文解析後に復元しなくてはいけない
  - 単一化文法の場合、確率計算のために遅延評価のメカニズムが必要

Shift-Reduce Parsing

- ２種類のデータ構造を持ち、２種類のアクションにより行う構文解析
  - スタック
    - 構文解析の途中結果を格納
  - キュー
    - まだ処理されていない単語列
  - Shift
    - キューの先頭の単語を取り出してスタックに積む
  - Reduce
    - スタックの上n個を取り出して文法規則を適用
    - 文法規則の適用結果(=親ノード)をスタックに積む

Shift-Reduce Parsing



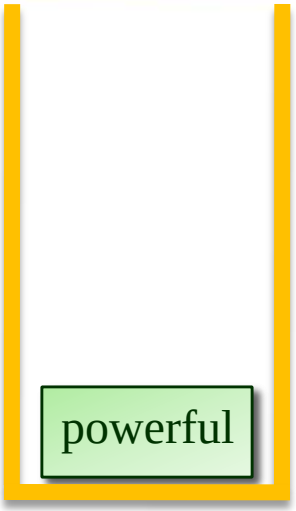
スタック

powerful	quake	injures	dozens	in	Japan

キュー

Shift!

Shift-Reduce Parsing



スタック

quake	injures	dozens	in	Japan

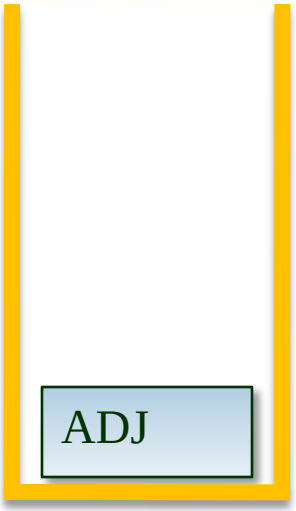
キュー

ADJ

|

powerful

Shift-Reduce Parsing



スタック

quake	injures	dozens	in	Japan

キュー

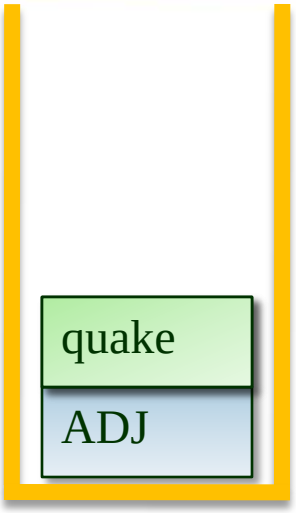
Shift!

ADJ

|

powerful

Shift-Reduce Parsing



スタック

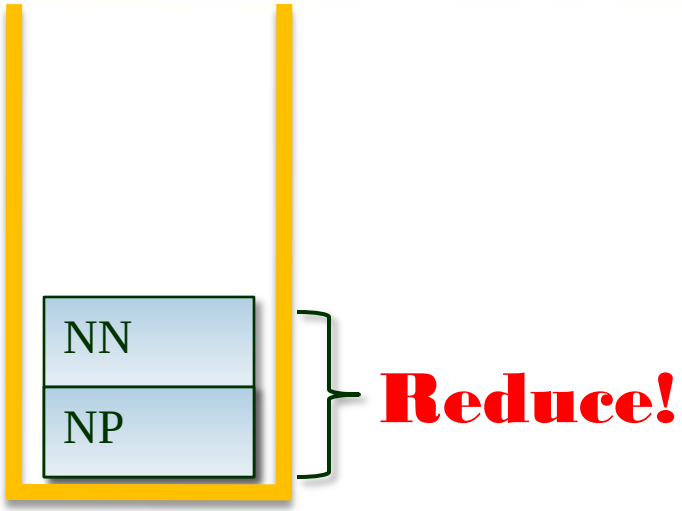
injures	dozens	in	Japan

キュー

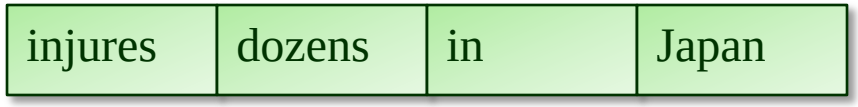
ADJ — powerful

NN — quake

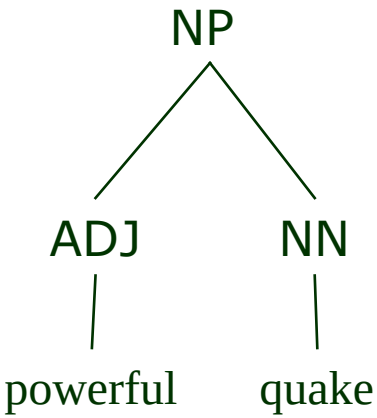
Shift-Reduce Parsing



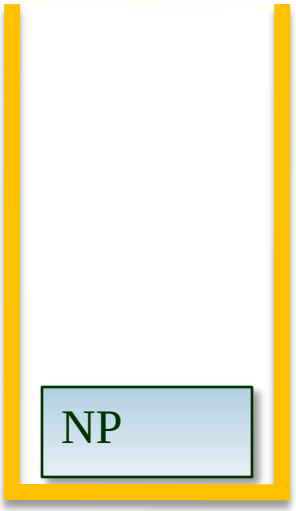
スタック



キュー



Shift-Reduce Parsing

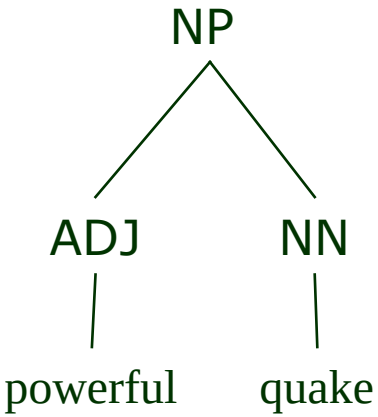


スタック

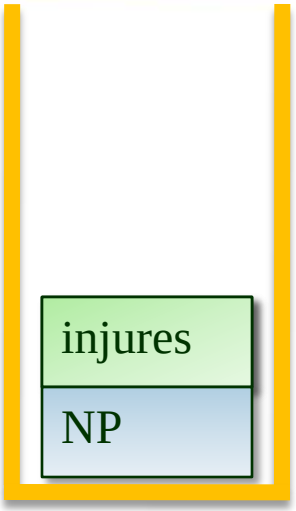
injures	dozens	in	Japan

キュー

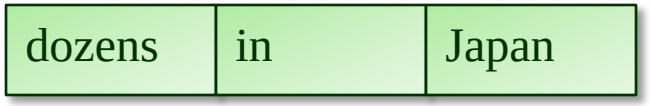
Shift!



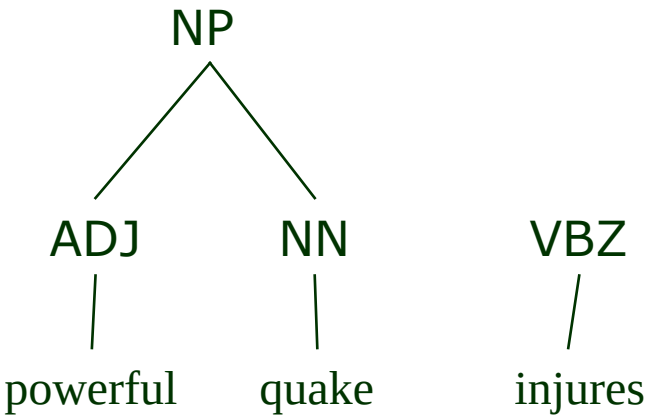
Shift-Reduce Parsing



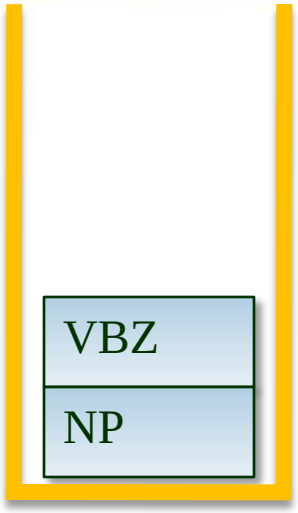
スタック



キュー



Shift-Reduce Parsing

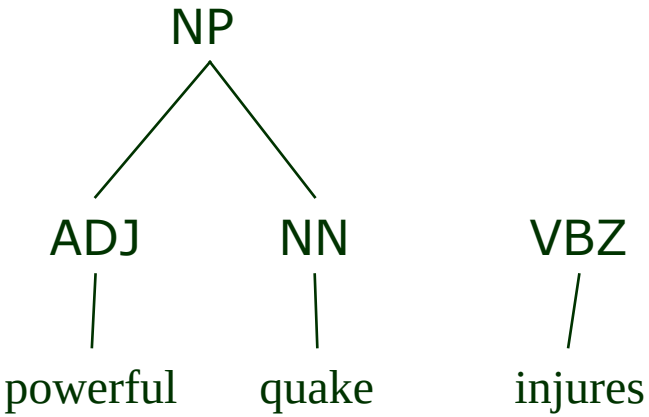


スタック

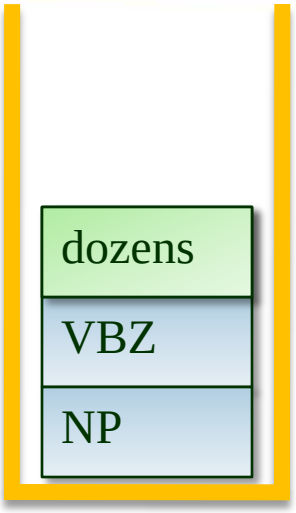
Shift!

dozens	in	Japan

キュー

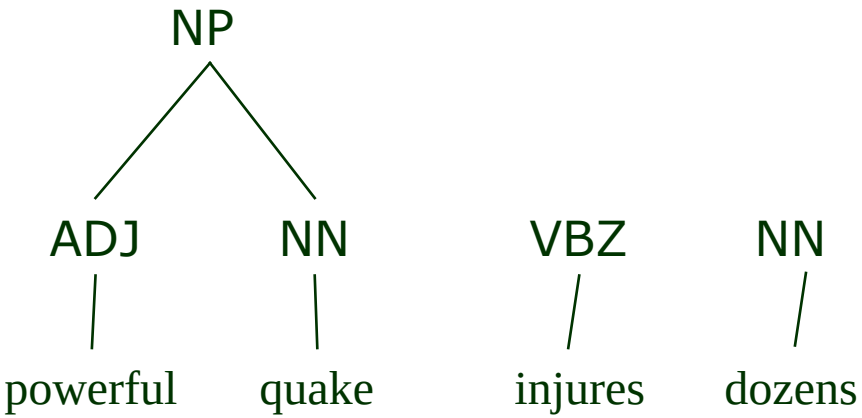


Shift-Reduce Parsing

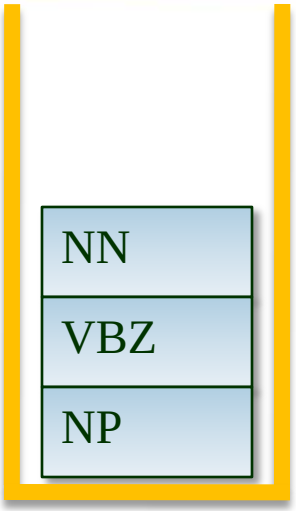


スタック

キュー



Shift-Reduce Parsing

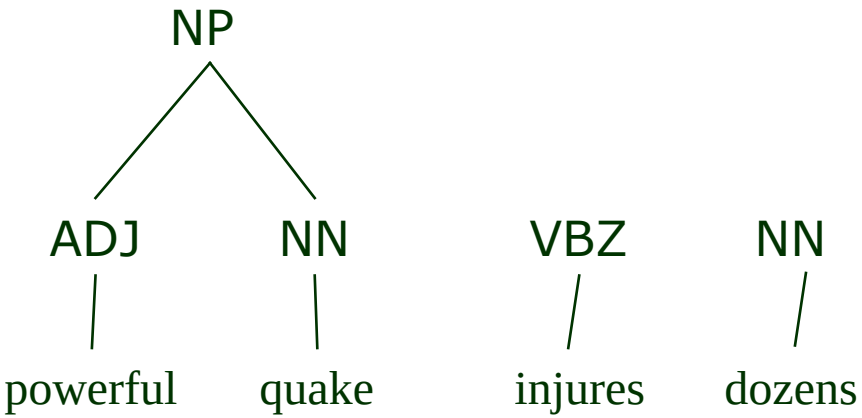


スタック

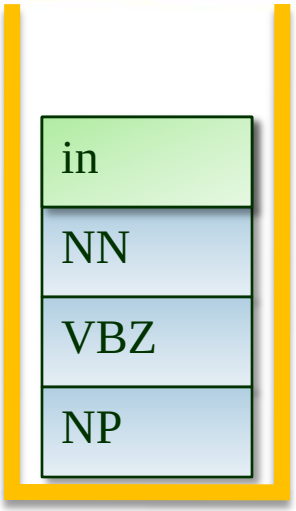
in	Japan

キュー

Shift!



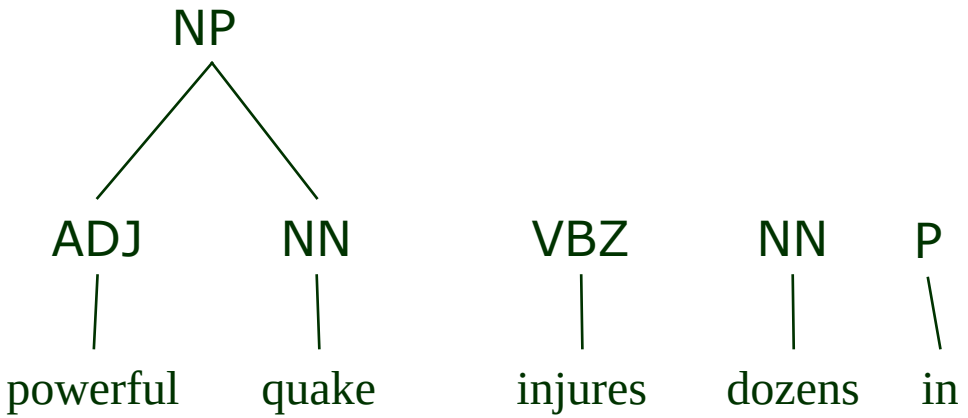
Shift-Reduce Parsing



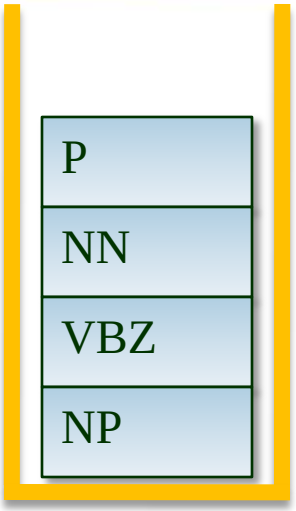
スタック

Japan

キュー



Shift-Reduce Parsing

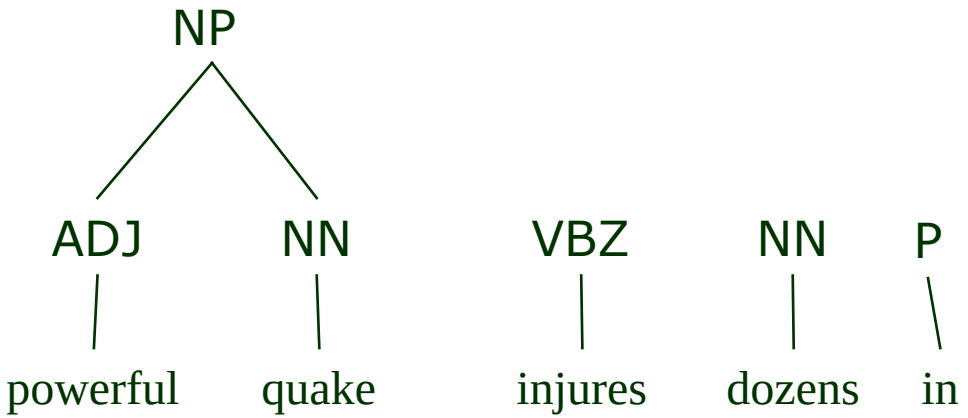


スタック

Shift!

キュー

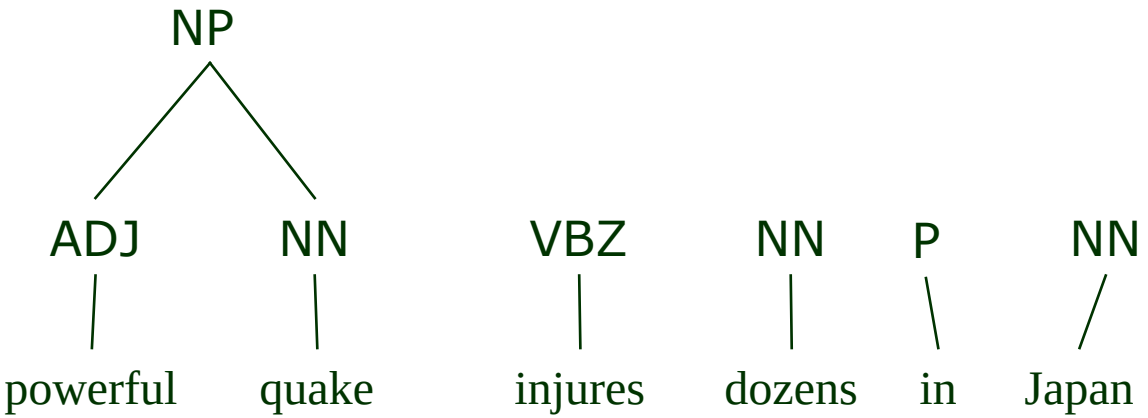
Japan



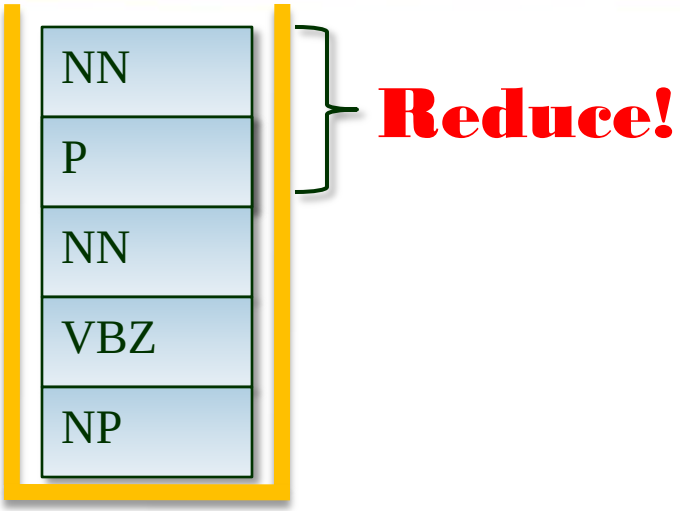
Shift-Reduce Parsing

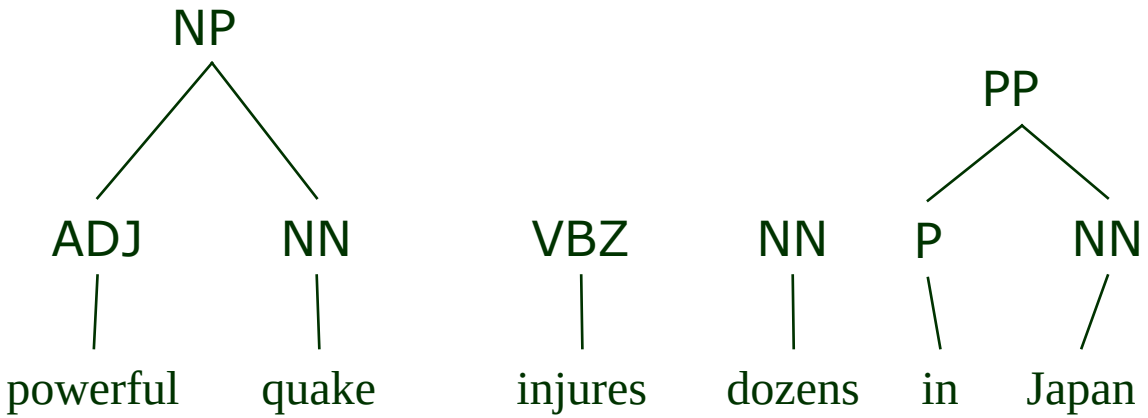
スタック

キュー

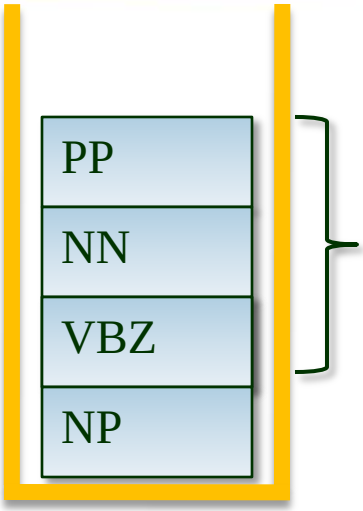


Shift-Reduce Parsing





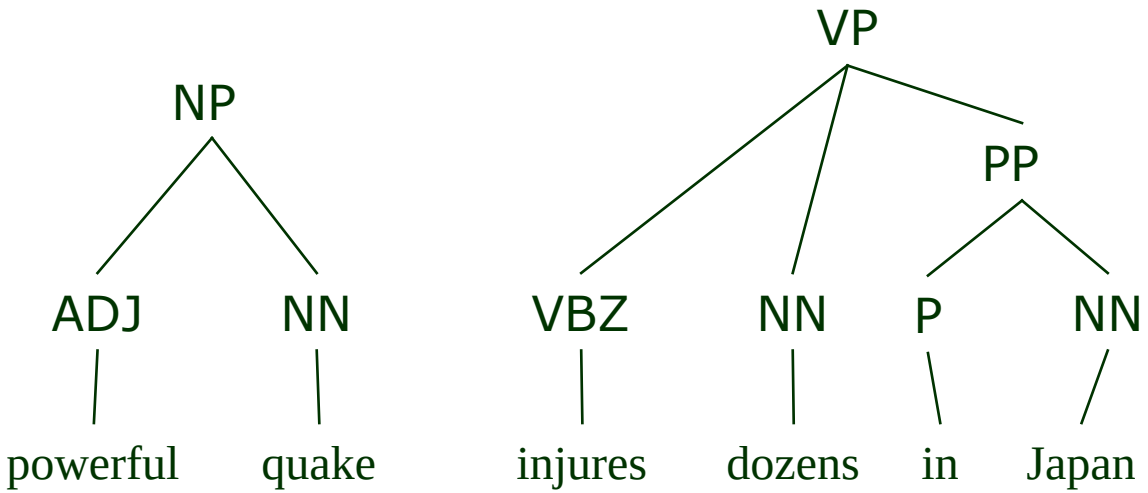
Shift-Reduce Parsing



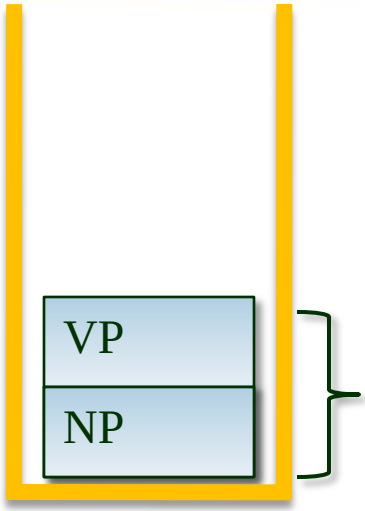
Reduce!

スタック

キュー



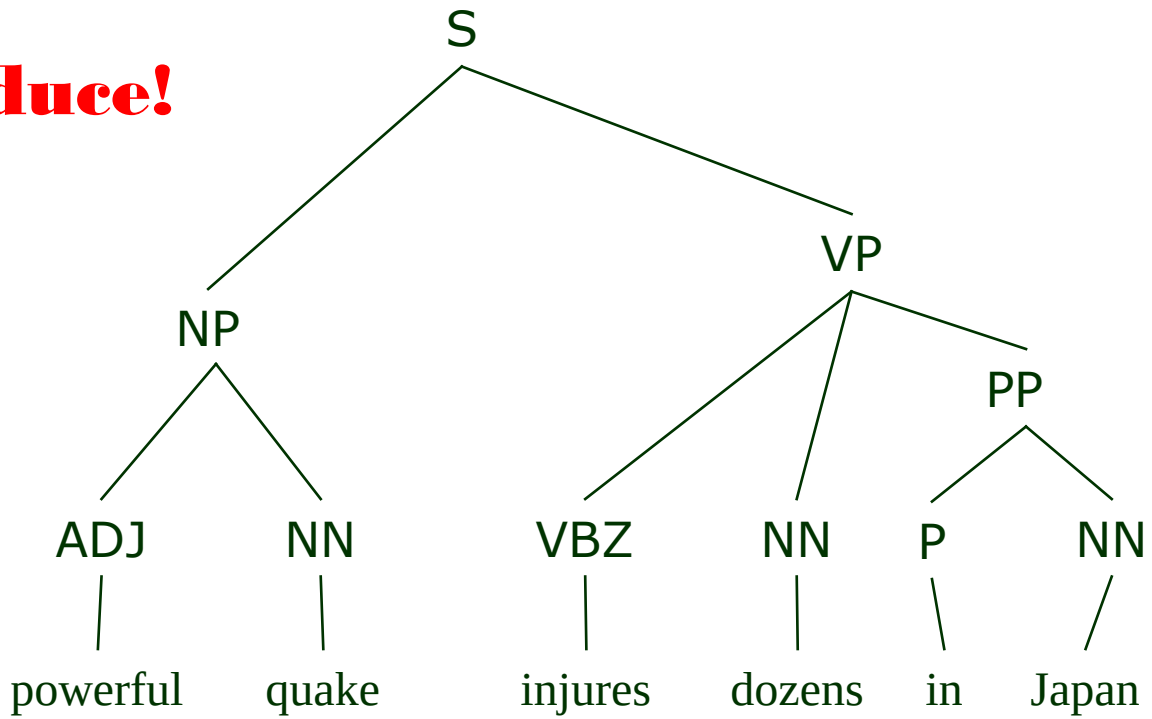
Shift-Reduce Parsing



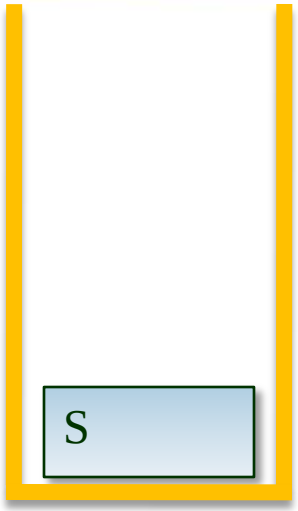
Reduce!

スタック

キュー



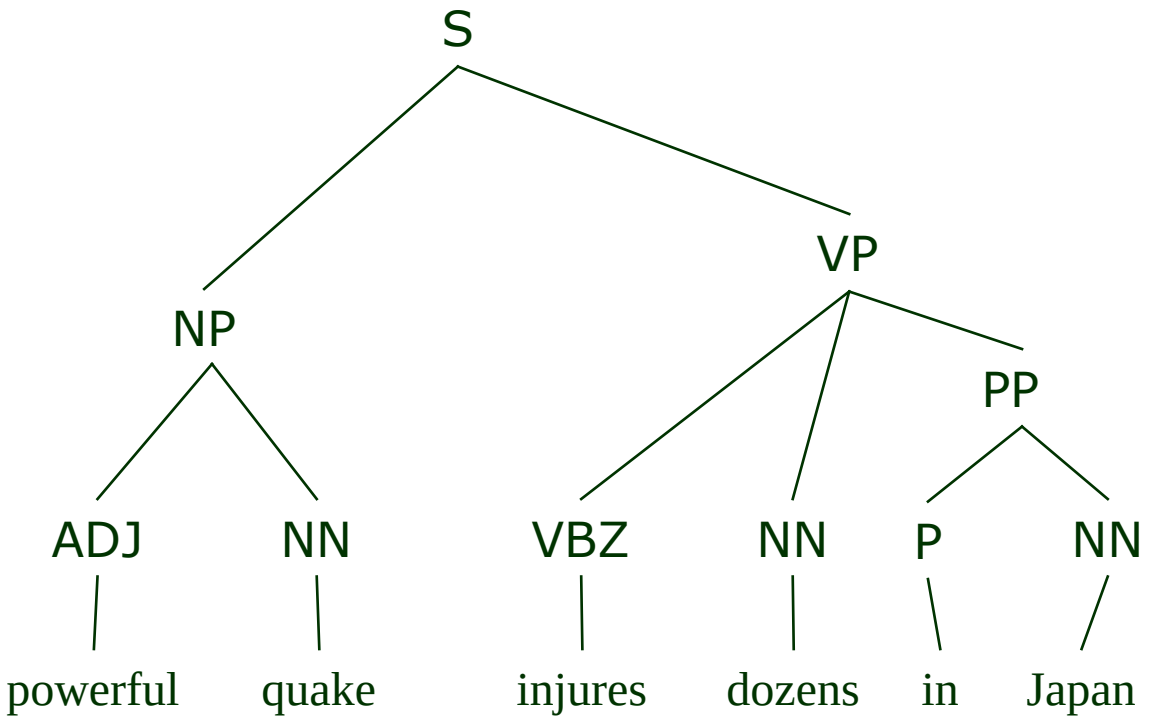
Shift-Reduce Parsing



スタック

完了！

キュー



Deterministic Shift-Reduce Parsing
(Ratnaparkhi1997, Yamada&Matsumoto2003, Sagae&Lavie2005)

- 識別器を用いて、どのアクションを行うか選択
  - SVMやMEを使う
  - 素性はスタック上の全ての構文木（大域素性）や単語、品詞（静的素性）
- 決定的に解析する（バックトラックを用いたり、複数の状態をもたない）

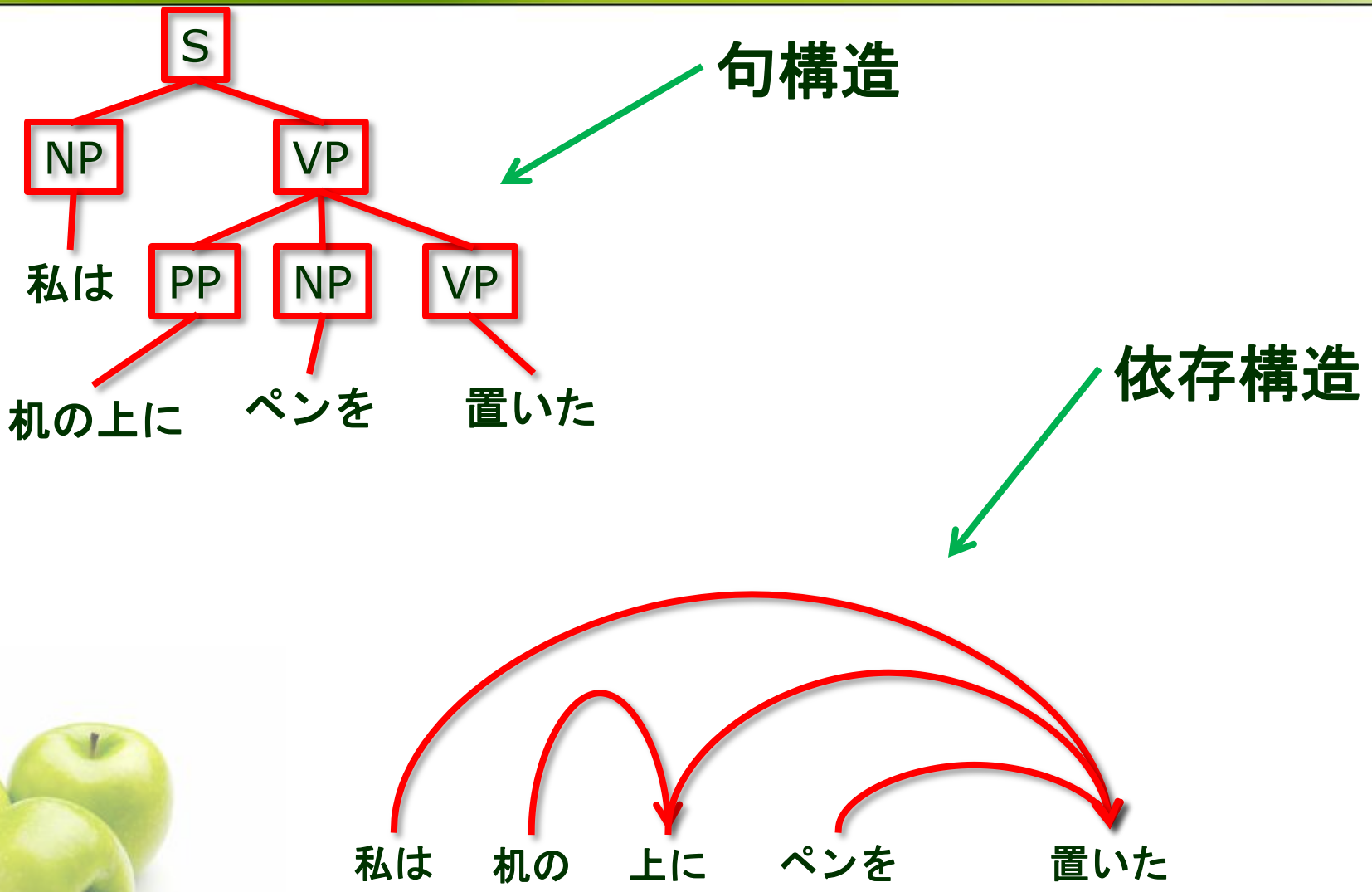
Shift-Reduce Parsing

- アクションの種類
  - Shift-X
    - Xという非終端記号(CFG)
    - Xという語彙項目(HPSG)
  - Reduce-Binary-X
    - 二分岐
    - Xという非終端記号が親ノード(CFG)
    - Xという文法規則(HPSG)
  - Reduce-Unary-X
    - 一分岐

MST Parsing for Dependency Analysis

MST法による依存構造解析

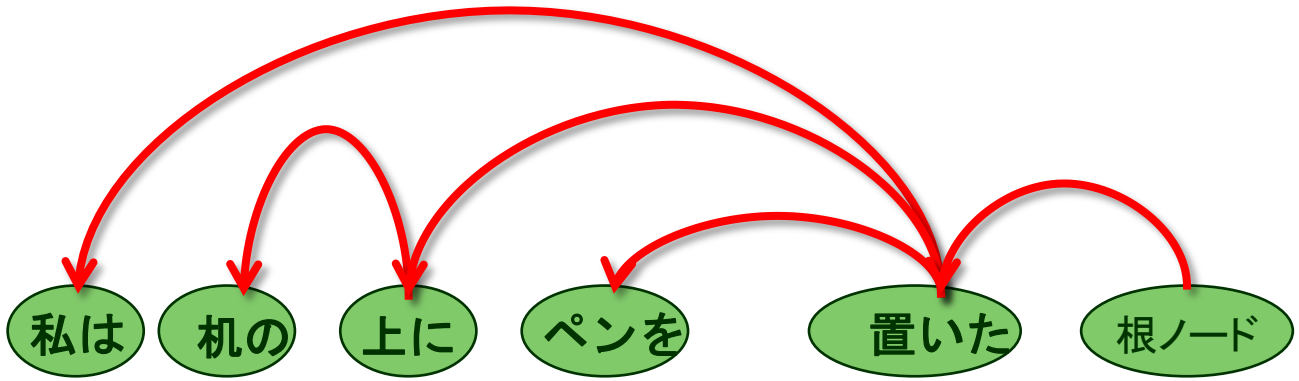
句構造と依存構造



依存構造の問題設定

- 有向木
- 有向木の根ノード=文全体を表す特殊なノード
- (根ノードを除いて)各単語は文中のどれかの単語一つに必ず係る

※有向木の弧の向きは、普通の係り受けの向きと逆になることに注意



依存構造解析

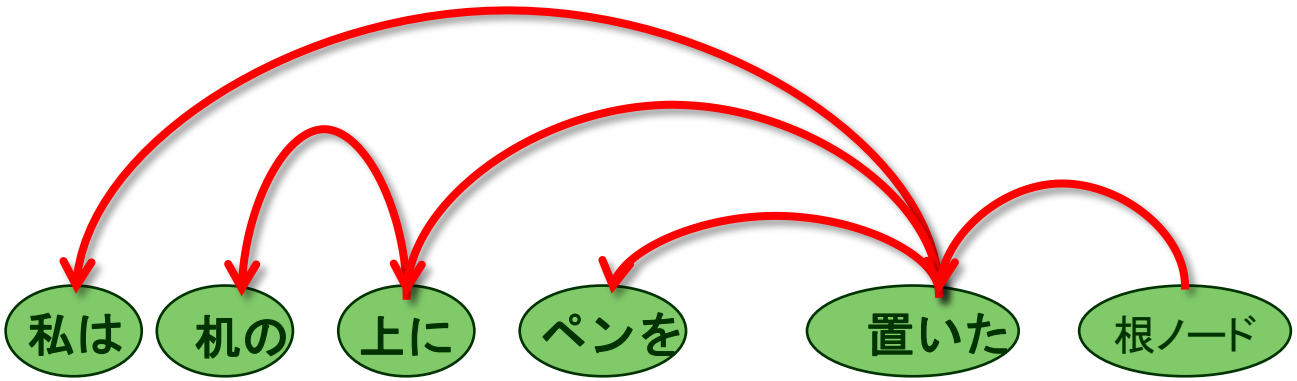
- cost(v, w): 単語wが単語vに係るコスト(有向木でのv→wの弧)

ある係り受けのスコア

$$\mathrm{cost}(Tree) = \sum_{\langle v,w\rangle \in Tree} \mathrm{cost}(v,w)$$

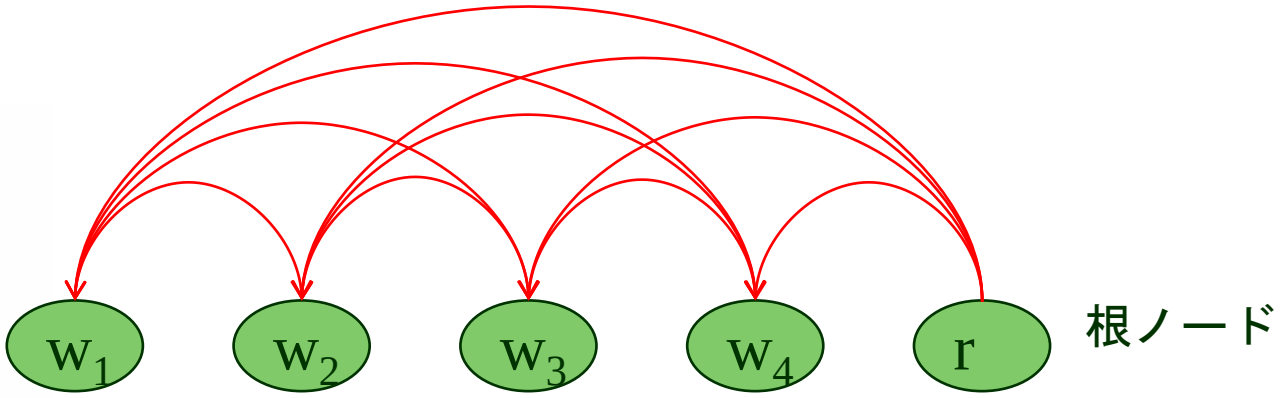
構文解析

$$\arg\min_{Tree} \sum_{\langle v,w\rangle \in Tree} \mathrm{cost}(v,w)$$



依存構造解析の問題設定

- 構文解析
  - 全ての単語v,wに対し、cost(v,w)が与えられていることを想定
  - 全てのノード間で弧が貼られた有向グラフから最もコストの低い有向木をみつける
  - 弧(v,w)の重みはcost(v,w)で与えられる



MST Parsingによる依存構造解析

- 構文解析
  - ⇒この問題は「最小コスト全域有向木問題(Minimum-Cost Arborescence Problem)」と等価

    (参考) 有名なMaximum Spanning Tree Problemは無向グラフであるときの最大全域木を求める問題
  - この問題は、**Chu–Liu/Edmonds's algorithm**で$O(n^2)$で解ける

Chu-Liu/Edmond's Algorithm

For each node $v \neq r$

Let y_v be the minimum cost of an edge entering node v

Modify the costs of all edges e entering v to cost(e)←cost(e)-y_v

Choose one 0-cost edge entering each $v \neq r$, obtaining a set F^*

If F^* forms an arborescence, then return it

Else there is a directed cycle $C \subseteq F^*$

Contract C to a single supernode, yielding a graph G'=(V', E')

Recursively find an optimal arborescence (V', F') in G'

Extend (V', F') to an arborescence (V, F) in G by adding all but one edge of C

(Algorithm Design, Jon Kleinberg & Eva Tardos, Pearson Education, 2006 より)

(アルゴリズムデザイン)

MST Parsingによる依存構造解析

- Perceptronによる学習
  - c_{ij}=cost(w_i, w_j)を学習する
  - あるコストc_{ij}が与えられた時コスト最小の依存構造を求めることができればより良いコストc_{ij}に更新できる

Input: training data D={<x,y>}, feature functions f={f_{ij}}, initial parameters c={c_{ij}}

Output: optimal parameters c

loop until c converges
  foreach <x,y> $\in$ D
    z' := argmin_z cost(z;x, c)
    if(y $\neq$ z')
      foreach f_{ij} $\in$ f
        c_{ij} := c_{ij} – f_{ij}(y) + f_{ij}(z')

まとめ

- さまざまな構文解析手法について紹介
  - CFGの条件付確率場
  - 決定性構文解析
  - 依存構造解析
- 講義資料
  - http://aiweb.cs.ehime-u.ac.jp/~ninomiya/ai2/

さいごに

- parsingの歌

  http://www.cs.jhu.edu/~jason/fun/grammar-and-the-sentence/