Basic issues on Parsing [1]

- Introduction
- Parsing issues
- Parsing context free grammar (CFG)
- Robust parsing
- Statistical parsing

Basic issues on Parsing $_2$

- Parsing goals
  - Syntactic structure
  - Semantic structure
- Syntax/semantic interaction
  - Only syntax
  - Only semantics
  - Performing in sequence
  - Performing in parallel

Basic issues on Parsing 3

- Parsing as searching in a search space
  - Characterizing the states
    - (if possible) enumerate them
  - Define the initial state (s)
  - Define (if possible) final states or the condition to reach one of them

Basic issues on Parsing 3

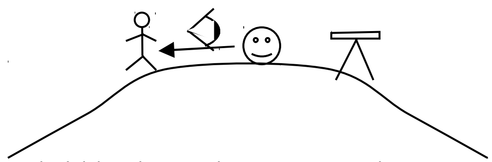
Ambiguity in parsing

A sentence is structurally ambiguos if the grammar assigns it more than a possible parse.

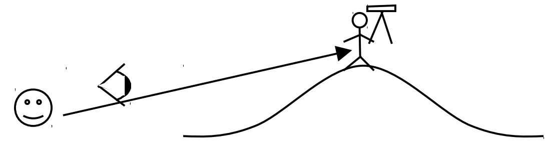
Common kinds of structural ambiguity include:

- PP-attachment
- Coordination ambiguity

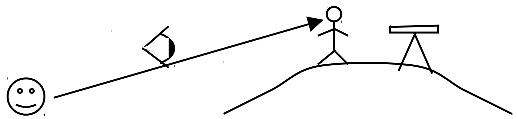
Basic issues on Parsing 5



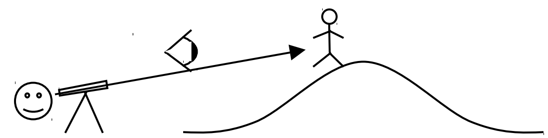
"I was on the hill that has a telescope when I saw a man."



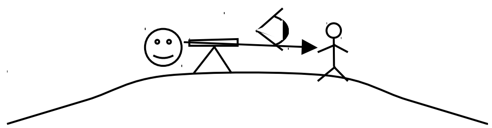
"I saw a man who was on a hill and who had a telescope."



"I saw a man who was on the hill that has a telescope on it."



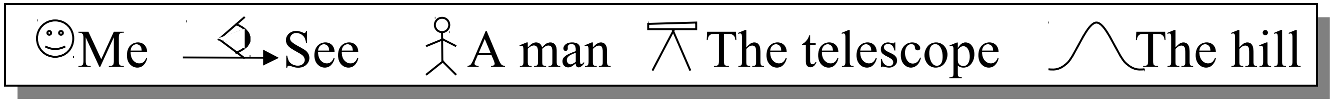
"Using a telescope, I saw a man who was on a hill."



"I was on the hill when I used the telescope to see a man."

. . .

I saw the man on the hill with the telescope



Taken from Loper

Basic issues on Parsing 4

Factors in parsing

- Grammar expressivity
- Coverage
- Involved Knowledge Sources
- Parsing strategy
- Parsing direction
- Ambiguity management
- (in)determinism
- Parsing engineering

Basic issues on Parsing 6

- Parsers today
  - Context free grammars (extended or not)
    - Tabular
    - Charts
    - Others
  - Unification-based
  - Statistical
  - Dependency parsing
  - Robust parsing (shallow, fragmental, chunkers, spotters)

Basic issues on Parsing 7

Parsing strategies

Parsing can be viewed as a search problem

Two common architectural approaches for this search are

Top-down: Starting with the root **S** and growing trees down to the input words

Bottom-up: Starting with the words and growing trees up

toward the root **S**.

Basic issues on Parsing 8

CFG grammar

Non terminal	Terminal
sentence → NP VP	det → the
NP → det n	n → cat
NP → n	n → fish
VP → vi	v → eats
VP → vt NP	

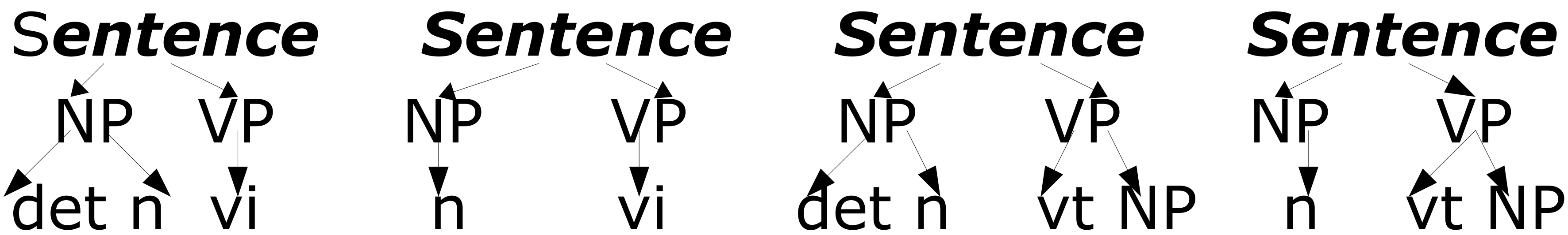
the cat eats fish

Basic issues on Parsing 9

Top-down

First step

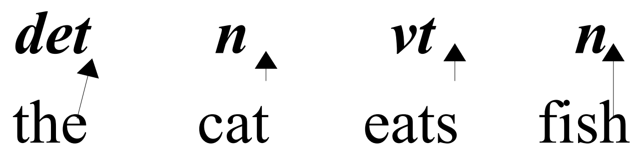
Second step



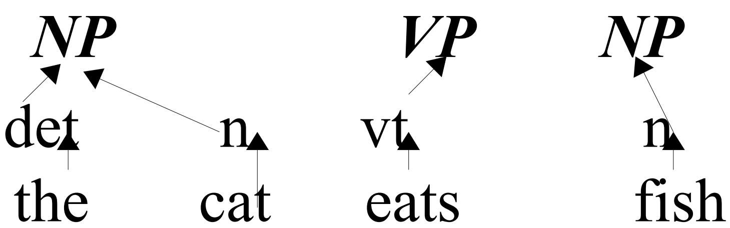
Basic issues on Parsing 10

Bottom-up

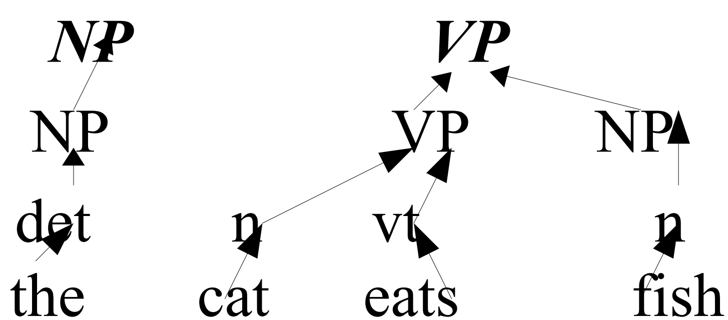
First step



Second step



Third step



Fourth step

Basic issues on Parsing 11

Parsing strategy

- **Top Down**
  - Guided by goals
  - Starts with a goal (or set of goals) to be built.
  - Tries to solve one of the pending goals
  - If more than one production can be applied:
    - serach problem
  - Pending goals can be reordered
  - Several search criteria (including heuristics) can be applied
  - The process ends when all the goals have been reached

Basic issues on Parsing 12

Parsing strategy

- **Bottom up**
  - Data driven
  - Starts from the sequence of words to be parsed (facts)
  - Proceeds bottom up
  - Several search criteria (including heuristics) can be applied
  - The process ends when the list of facts contains the initial symbol of the grammar

Basic issues on Parsing 13

- Problems of ***top down*** strategy
  - Left recursivity
  - Many productions expanding the same non terminal
  - Useless work
  - Search basically guided by the grammar
  - Repeated work
  - Problems of backtracking algorithms

Basic issues on Parsing 14

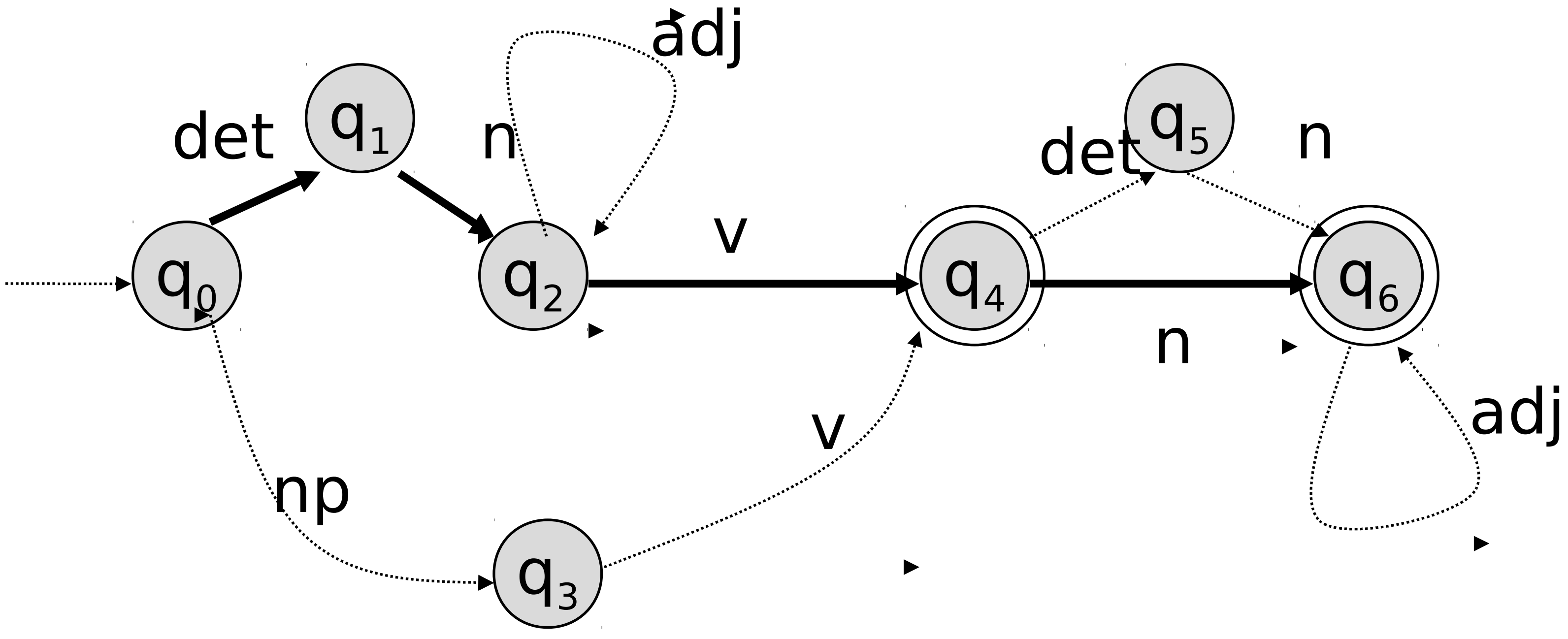
- Problems of ***bottom up*** parsing
  - empty (optional) categories
  - Useless work (locally possible but globally impossible)
  - Inefficient when there is a high lexical ambiguity
  - Repeated work

Transition Networks 1

Finite state automata->Transition Network (TN)

- States associated to the positions in the sentence
- Arcs (transitions)
  - Labeled with part of speech (POS)
    - An arc can be traversed if the current word has the same POS as the arc.
- Non determinism
  - More than one initial state
  - Current word with more than 1 POS
  - More than one arc for the same POS

Transition Networks $_2$



the *cat* *eats* *fish*
det n v n

Transition Networks 3

Transition networks limitations

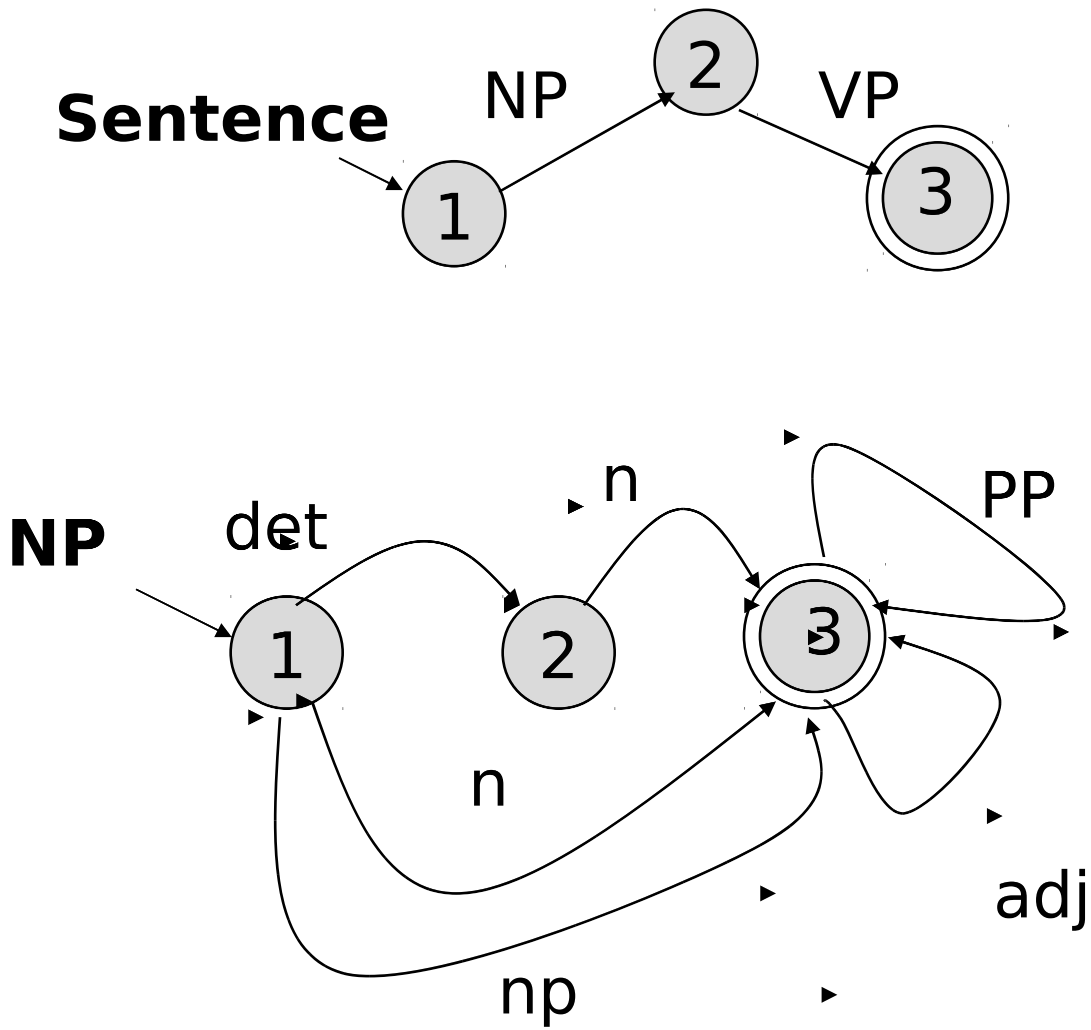
- Only regular grammars
- Only recognition
- Non-determinism -> backtracking
- No separation between grammar and parser
  - grammar -> syntactic model description
  - parser -> control

Transition Networks 4

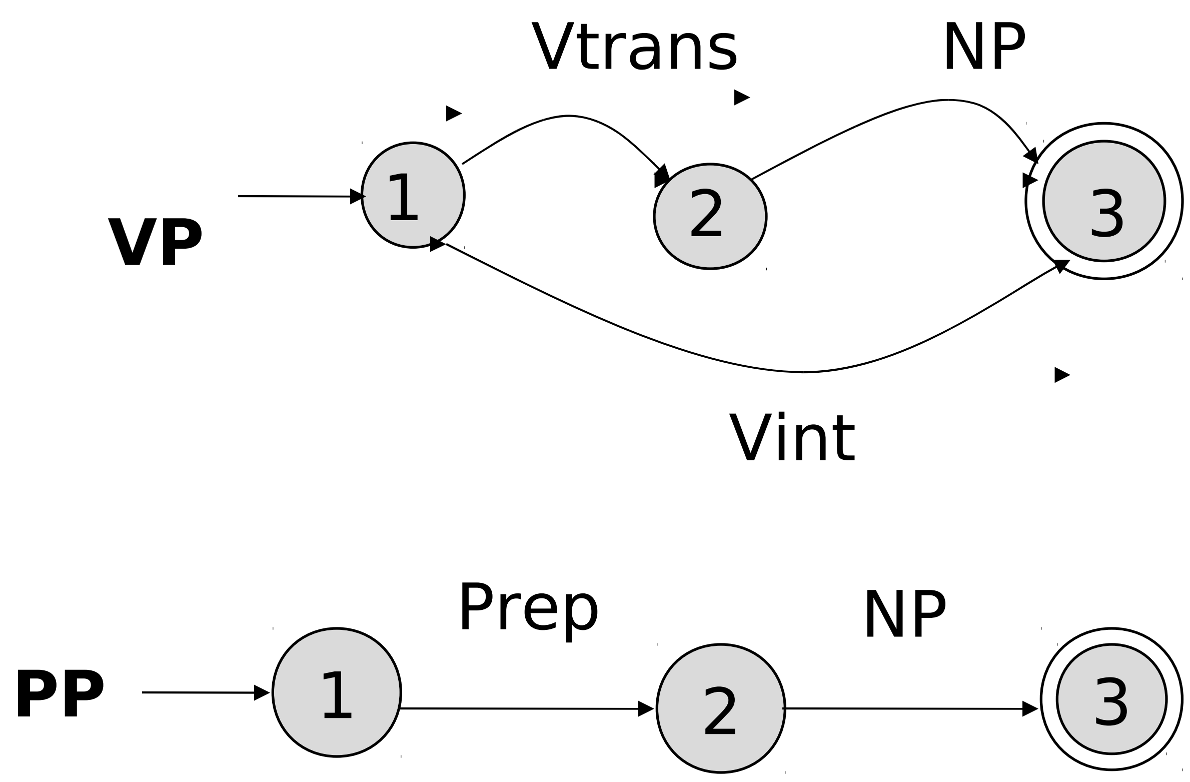
Recursive transition networks (RTN)

- Colection of TNs labeled with a name
  - Arcs
    - Labeled as in TN with POS
      - Terminal labels
    - Labeled with RTN identifiers
      - Non terminal labels
      - Final states in RTN produce coming back to the target state of the arc producing the call
- RTN are weakly equivalent to CFG

Transition Networks 5



Transition Networks 6



Transition Networks 7

Recursive transition networks (RTN) Limitations

- Transitions depend only on the categories
  - CFG
- Only recognizing
- In fact fixed top-down strategy

Transition Networks 8

- Woods (1970)
- **Aumented transition networks (ATN)** = RTN with *operations* attached to arcs and use of *registers*

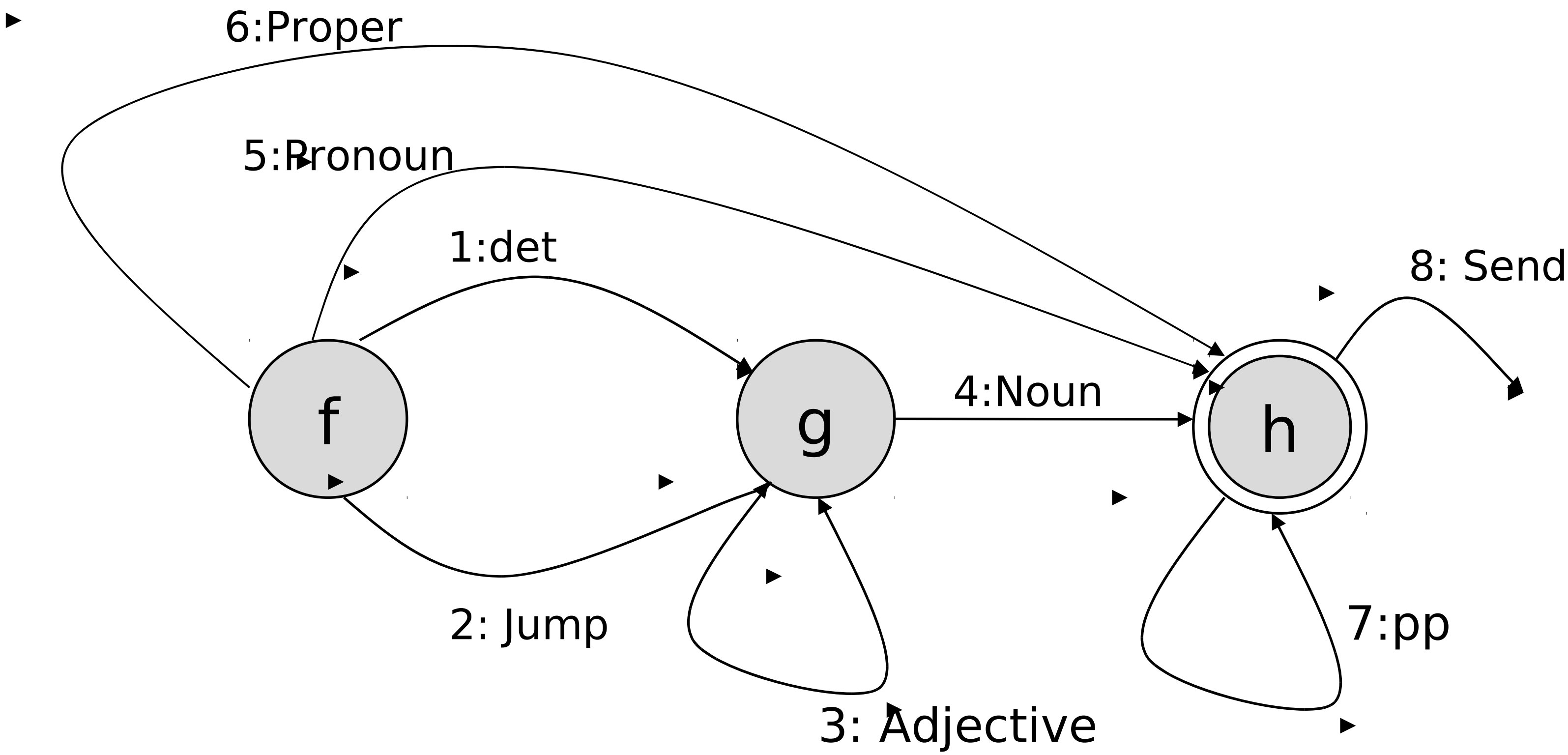
Operations

Conditions
Filter transitions between states
Actions
Building intermediate and output structures.
Initializations

- ATN allow expressing contextual constraints

Transition Networks 9

Features
Number: Singular, Plural   Default: empty
Person: 1st, 2nd, 3rd   Default: 3rd
Rols: Subject



Taken from Winograd, 1983

Transition Networks 10

Inicializations, Conditions and Actions

NP-1: $_f$Determiner$_g$

A: Set Number to the number of *

NP-4: $_g$Noun$_h$

C: Number is empty or number is the number of *

A: Set Number to the number of *

Set Subject to *

NP-5: $_f$Pronoun$_h$

A: Set Number to the number of *

Set Person to the Person of *

Set Subject to *

NP-6: $_f$Proper$_h$

A: Set Number to the number of *

Set Subject to *

Transition Networks 11

Aumented Transition networks limitations

- Fixed top-down strategy
- Redundancy in backtracking operations
- Problems of notational expressivity:
  - Very difficult to transport

Basic issues on Parsing 11

- Unified mechanism of parser description
  - Sikkel, 1997
- Parser (schema):
  - Given a sentence, an inicial set of items is build
  - Given a grammar, a set of rules can be used for getting additional items
- Parser (algorithm):

  Parsing schema

  + data structures

  + control structures

  (+ communication structures)

Charts$_1$

- A *Chart* is a directed graph built dynamically along parsing
- Extension of WFST
- Nodes correspond to the start and end of the sentence and to the positions between words.
- Active arcs (goals or hypothesis) and inactive arcs (facts)
  - Notation active arcs: dotted rules
  - inactive arcs : category

Charts$_2$

CFG grammar

Non terminal	Terminal
sentence → NP VP	det → the
NP → det n	n → cat
NP → n	n → fish
VP → vi	v → eats
VP → vt NP	

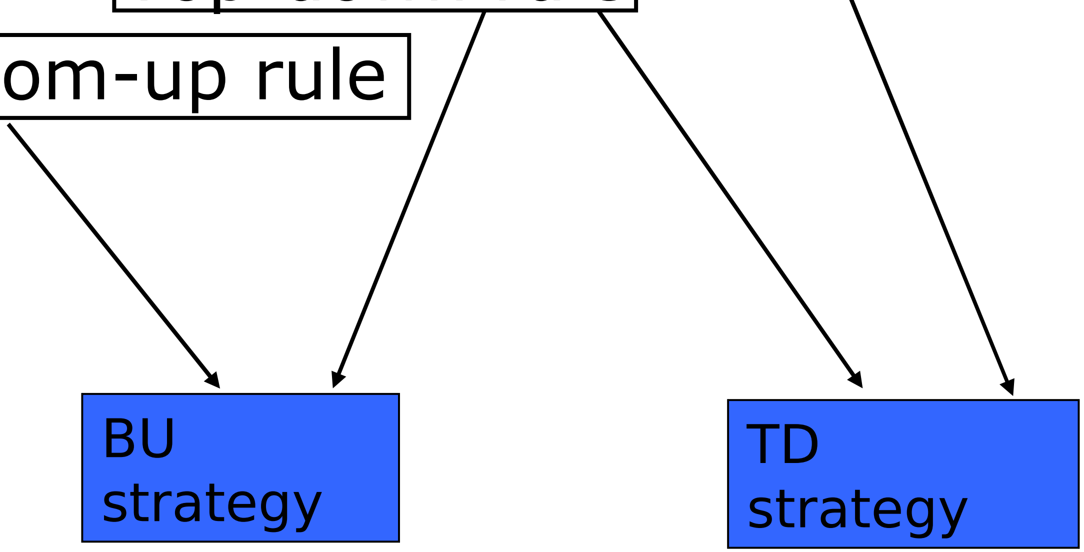
the cat eats fish

Charts$_3$

```
program chart
{ inicialize the chart with H;
  inicialize the agenda with items which can be deduced without antecedents;
  while not empty (agenda)
  {extract current_item from agenda and put it on the chart;
    foreach item which could be deduced with one step including current_item
     {if item not in  agenda and not in chart
      then add item to agenda
     }
  }
}
```

Charts$_4$

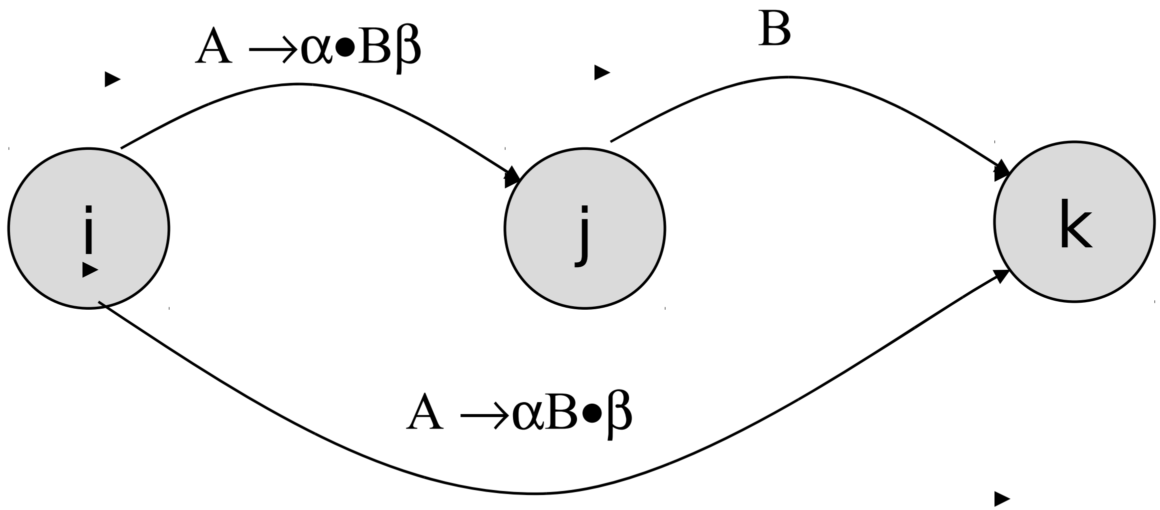
- A concrete chart algorithm should:
  - define the structure of *agenda* and its scheduling criteria
  - define order of performing deductive steps
- $D^{scan} + D^{compl}$ -> Combination rule
- D^{pred} -> Top-down rule
- Bottom-up rule



Charts$_5$

Combination rule

When an active arc of the Chart reaches a node ***j*** and from this node starts an
inactive arc labeled with the category the active arc was waiting for, both arcs are combined for building a new arc (active or not) starting in the start node of the active arc and ending in the ending node of the inactive arc.

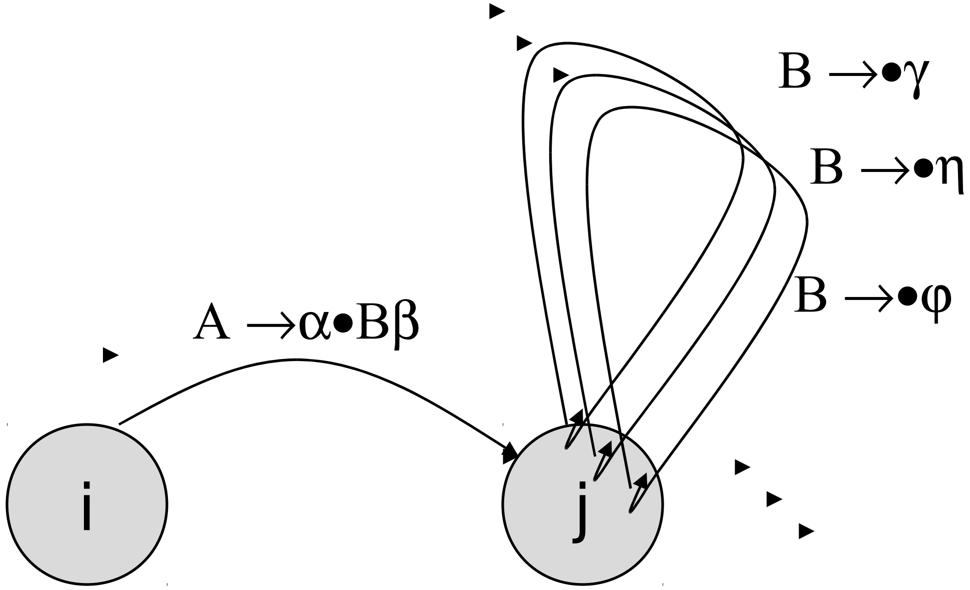


Charts$_6$

Top-down rule

When an active arc of the Chart reaches a node ***j***, for all the productions of the grammar expanding the category the active arc is waiting for a new active arc is built starting and ending in ***j*** corresponding to the dotted rule with dot in the initial position.

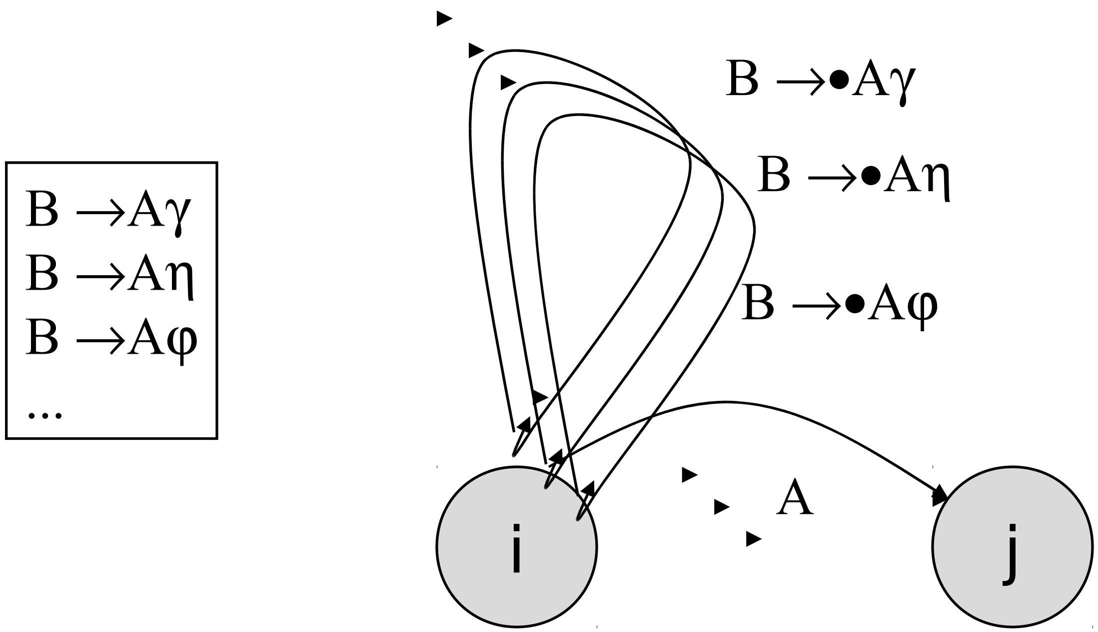
B →γ
B →η
B →φ
...



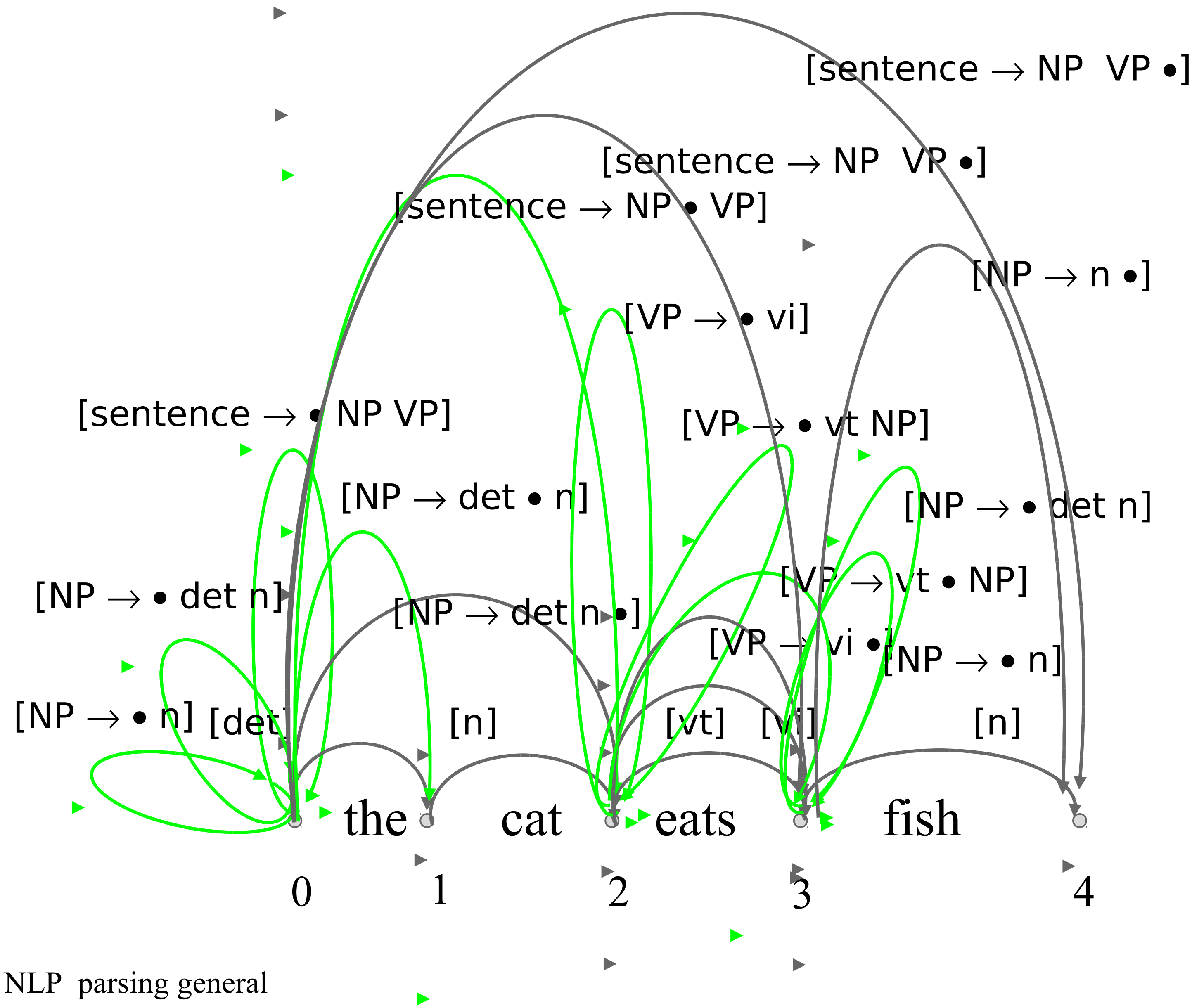
Charts$_7$

Bottom-up rule

When an inactive arc of the Chart starts in a node ***i***, for each Producction of the grammar owning as first copnstituent of the right side the category of the inactive arc a new active arc is built starting and ending in ***i*** corresponding to the dotted rule with dot in the initial position



Charts$_8$



[sentence → NP VP •]
[sentence → NP VP •]
[sentence → NP • VP]
[NP → n •]
[VP → • vi]
[sentence → • NP VP]
[VP → • vt NP]
[NP → det • n]
[NP → • det n]
[NP → • det n]
[VP → vt • NP]
[NP → det n •]
[VP → vi •]
[NP → • n]
[NP → • n]
[det]
[n]
[vt]
[vi]
[n]
the
cat
eats
fish
0
1
2
3
4

Charts$_9$

- Problems
  - The size of the chart grows with the size of the grammar making the algorithm difficult to scale up.
  - A lot of useless active and inactive arcs are built.
  - In practice, lacking appropriate knowledge, a fixed bottom-up strategy, eventually corrected with top-down predictions, is used

Charts$_{10}$

- Ambiguity combined with the repeated parsing of sub-trees are a difficulty for parsing algorithms. Those algorithms use simple backtracking mechanisms.

- There parsing algorithms that use dynamic programming techniques, such as a table of partial parsers to efficiently parse ambiguous sentences.

-The CKY, Earley and Chart-Parsing algorithms all use dynamic-programming to solve repeated parsing of subtrees problem.

Robust Parsing$_1$

- Conventional methods of parsing are insufficient for dealing with non restricted texts
  - Adequate segmentation → Which are the units to parse
  - Disambiguation → Which is the most likely parse
  - Coverage → Parsing beyond the lexical coverage

- What to do
  - Not parsing all → fragmental parsing
  - Not parsing in depth → shallow parsing

Robust Parsing$_2$

- Problems when parsing non restricted corpus
- Adaptation of a grammar to a corpus or sublanguage
- Selection of the correct (!?) parse between the ones allowed by the grammar
- Production of good parses for entries outside the coverage of the grammar (Robustness)

Robust Parsing$_2$

- Partial parsing and chunking are methods for identitying shallow syntactic constituents in a text.

- High accuracy partial parsing can be achieved either trough rule-based or machine learning-based methods.

Robust Parsing$_3$

Partial parsers

phrasal parsers

- chunkers, spotters
- Church,1988

coocurrence parsers

- Church,Hanks,1989, Brent,1993

fragmental parsers

- Fidditch, Hindle,1994, MITFP, Abney,1991

constraint-based parsers

- Voutilainen,1995

Robust Parsing$_4$

- Chunking
  - detection of phrases nominal, verbal, adjetival, adverbial basic (without recursion)
  - Finite state techniques
  - Performance of a cascade of transductors
  - Hidden Markov Models
  - Machine Learning

Abney, 1996
Argamon et al, 1998
Cardie, Pierce, 1998
Church, 1988
Ramshaw, Marcus, 1995
Skut, Brants, 1998

Robust Parsing$_5$

Definition of chunk

With linguistic basis: Abney

Only pragmatic:

- Contiguous sequences of related tokens
  - Not confusing with terms
- e.g. Base NP

Approaches to chunking

Look for (include) information

Remove information

- e.g. Chink

Robust Parsing$_6$

Representing chunks

Labels

- e.g. tags
- BEGIN, INSIDE, OUTSIDE

Trees

Chunk parser

Looking for non overlapped chunks for reaching a maximal coverage

Robust Parsing$_7$

Frecuently regular expressions over sequences of POS tags

aglomerative (chunk rules) vs divisive (chink rules)

- Rules for fusion of adjacent chunks
- Rules for splitting a chunk in smaller components

Statistical Parsing$_1$

Using **statistical models** for

- Guiding parsing
  - Get the most likely parse
  - Ambiguity resolution (pp-attachment)
  - Grammatical induction from corpora

Goal: Parsing of non restricted texts with a reasonable level of accuracy (>90%) and efficiency.

Requirements:

Corpora tagged (with POS): Brown, LOB, Clic-Talp

Corpora analyzed: Penn treebank, Susanne, Ancora

Statistical Parsing$_2$

- Lexical approaches
  - Context free: unigram
  - Context dependent: N-gram, HMM

- Syntactic approaches
  - Statistical context free grammar (SCFG)

- Hibrid approaches
  - Stochastic lexicalized Tags
  - Computing the most likeky (most probable) parse

Statistical Parsing$_3$

Probabilistic grammars assign a probability to a sentence or string of words.

Usually they capture more general syntactic information than the N-gram grammars.

In a probabilistic context-free grammar (PCFG):

- Associate a probability to each rule
- Associate a probability to each lexical entry

Each PCFG rule is treated as if it were conditionally independent; thus the probability of a sentence is computed by multiplying the probabilities of each rule in the parse of the sentence.

Statistical Parsing$_4$

S	$\Rightarrow$	NP	VP	1.0
VP	$\Rightarrow$	Vi		0.4
VP	$\Rightarrow$	Vt	NP	0.4
VP	$\Rightarrow$	VP	PP	0.2
NP	$\Rightarrow$	DT	NN	0.3
NP	$\Rightarrow$	NP	PP	0.7
PP	$\Rightarrow$	P	NP	1.0

Vi	$\Rightarrow$	sleeps	1.0
Vt	$\Rightarrow$	saw	1.0
NN	$\Rightarrow$	man	0.7
NN	$\Rightarrow$	woman	0.2
NN	$\Rightarrow$	telescope	0.1
DT	$\Rightarrow$	the	1.0
IN	$\Rightarrow$	with	0.5
IN	$\Rightarrow$	in	0.5

- Probability of a tree t with rules

$$\alpha_1 \to \beta_1, \alpha_2 \to \beta_2, \ldots, \alpha_n \to \beta_n$$

is

$$p(t) = \prod_{i=1}^{n} q(\alpha_i \to \beta_i)$$

where $q(\alpha \to \beta)$ is the probability for rule $\alpha \to \beta$.

Statistical Parsing$_5$

1. A context-free grammar $G = (N, \Sigma, S, R)$.

2. A parameter

$$q(\alpha \rightarrow \beta)$$

for each rule $\alpha \rightarrow \beta \in R$. The parameter $q(\alpha \rightarrow \beta)$ can be interpreted as the conditional probabilty of choosing rule $\alpha \rightarrow \beta$ in a left-most derivation, given that the non-terminal being expanded is α. For any $X \in N$, we have the constraint

$$\sum_{\alpha \rightarrow \beta \in R: \alpha = X} q(\alpha \rightarrow \beta) = 1$$

In addition we have $q(\alpha \rightarrow \beta) \geq 0$ for any $\alpha \rightarrow \beta \in R$.

Given a parse-tree $t \in \mathcal{T}_G$ containing rules $\alpha_1 \rightarrow \beta_1, \alpha_2 \rightarrow \beta_2, \ldots, \alpha_n \rightarrow \beta_n$, the probability of t under the PCFG is

$$p(t) = \prod_{i=1}^{n} q(\alpha_i \rightarrow \beta_i)$$

Statistical Parsing$_6$

- Assigns a probability to each *left-most derivation*, or parse-tree, allowed by the underlying CFG

- Say we have a sentence s, set of derivations for that sentence is $\mathcal{T}(s)$. Then a PCFG assigns a probability $p(t)$ to each member of $\mathcal{T}(s)$. i.e., *we now have a ranking in order of probability*.

- The most likely parse tree for a sentence s is

$$\arg\max_{t \in \mathcal{T}(s)} p(t)$$

Statistical Parsing, 7

- **Learning.** How to obtain a PCFG from a treebank?
- **Inference.** Given a PCFG and a sentence s. Denote by $\mathcal{T}(s)$ the set of derivations that yield s.
  - How to compute the best parse for s?

$$\operatorname*{argmax}_{t \in \mathcal{T}(s)} p(t)$$

  - How to compute the probability of s?

$$\sum_{t \in \mathcal{T}(s)} p(t)$$

Statistical Parsing$_7$

Pros and cons of SCFG

Some idea of the probability of a parse but not very good

CFG cannot be learned without negative examples, SCFG can SCFGs provide a language model for a language

In practice SCFG provide a worse language model than a 3-gram

P([N [N toy] [N [N coffee] [N grinder]]]) =

P ([N [N [N cat] [N food]] [N tin]])

P (NP → Pro) is > in Subj position than in Obj position.

Statistical Parsing$_8$

- Robust
- Posibility of combining SCFG with 3-grams
- SCFG assign a lot of probability mass to short sentences (a small tree is more probable than a big one)
- Parameter estimation (probabilities)
- Problem of sparseness
- Volume

Statistical Parsing$_9$

- Association to the rule. Information about the point of application of the rule in the derivation tree is lost.
- Low frequency constructions are penalized
- Probability of a derivation. Contextual independence is assumed
- Posibility of relax conditional independence:
  - Sensitivity to structure
  - Lexicalization

Statistical Parsing$_{10}$

Sensitivity to structure

Node expansion depends of its position in the tree

	Pronoun	Lexical
Subject	91%	9%
Object	34%	66%

Enrichment of a node with information of its ancestors

- ^{S}NP is different of ^{VP}NP

Pronouns as arguments of ditransitive verbs

- *I gave Charlie the book*
- *I gave the book to Charlie*
- *I gave you the book*
- *? I gave the book to you*

Statistical Parsing$_{11}$

(Head) Lexicalization

put takes both an NP and a VP

- *Sue put* [*the book*]$_{NP}$ [*on the table*]$_{PP}$
- * *Sue put* [*the book*]$_{NP}$
- * *Sue put* [*on the table*]$_{PP}$

like usually takes an NP and not a PP

- *Sue likes* [*the book*]$_{NP}$
- * *Sue likes* [*on the table*]$_{PP}$

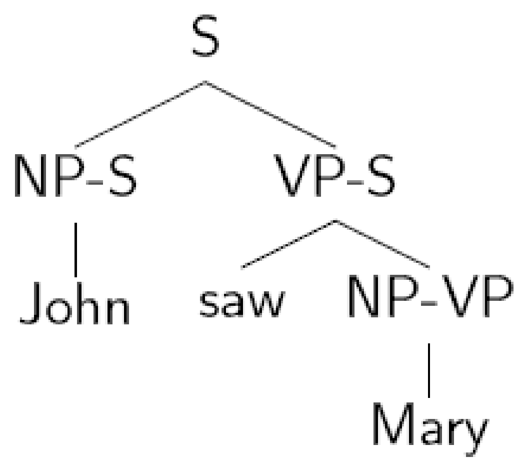
Statistical Parsing$_{12}$

Local Tree	*Come*	*Take*	*Think*	*Want*
VP-> V	9.5%	2.6%	4.6%	5.7%
VP-> V NP	1.1%	32.1%	0.2%	13.9%
VP-> V PP	34.5%	3.1%	7.1%	0.3%
VP- V SBAR	6.6%	0.3%	73.0%	0.2%
VP-> V S	2.2%	1.3%	4.8%	70.8%
VP->V NP S	0.1%	5.7%	0.0%	0.3%
VP->V PRT NP	0.3%	5.8%	0.0%	0.0%
VP->V PRT PP	6.1%	1.5%	0.2%	0.0%

Statistical Parsing$_{13}$

PCFGs with Parent Annotations

(Johnson, 1999)

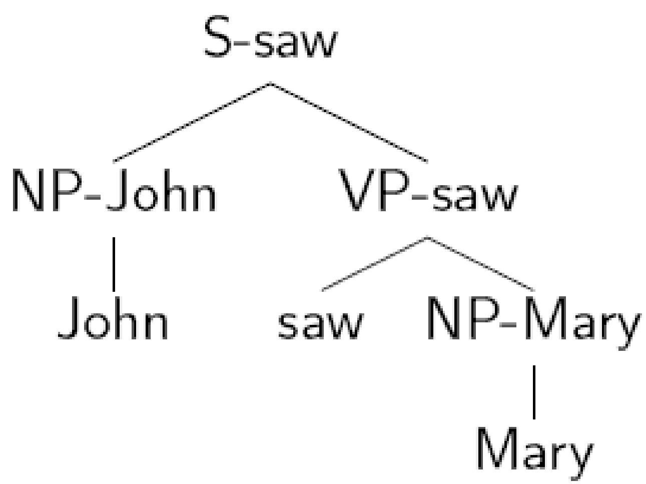


$$
\begin{aligned}
P(Tree) \;=\; & q(\texttt{S -> NP-S VP-S} \mid \texttt{S}) \times \\
& q(\texttt{NP-S -> John} \mid \texttt{NP-S}) \times \\
& \ldots
\end{aligned}
$$

Statistical Parsing$_{14}$

Lexicalized PCFGs

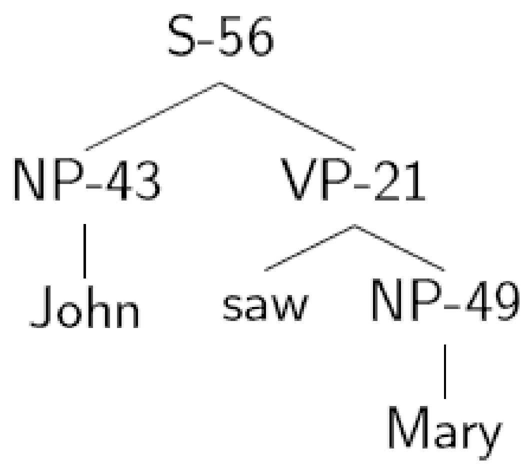
(Collins, 1999)



$$
\begin{aligned}
P(Tree) \quad = \quad & P(\texttt{S-saw -> NP-John VP-saw} \mid \texttt{S-saw}) \times \\
& P(\texttt{NP-John -> John} \mid \texttt{NP-John}) \times \\
& \ldots
\end{aligned}
$$

PCFGs with Latent Variables

(e.g., Petrov and Klein, 2007)



- Each non-terminals (e.g., S) is split into a number of new non-terminals (e.g., S−1, S−2, . . ., S−128)
- Latent annotations learned using EM

Statistical Parsing$_{15}$

Two models

Conditional/Discriminative model :

The probability of a parse tree is directly estimated

$$P(t|s, G) \text{ con } \sum_t P(t|s, G) = 1$$

Probabilities are conditioned on a concrete sentence.

No sentence distribution probabilities is assumed

Statistical Parsing$_{16}$

- CFG
- SCFG
  - For each rule of G, $(A \rightarrow \alpha) \in P_G$ we should be able to define a probability $P(A \rightarrow \alpha)$

$$\sum_{(A \rightarrow \alpha) \in P_G} P(A \rightarrow \alpha) = 1$$

- Probability of a tree

$$P(\psi) = \prod_{(A \rightarrow \alpha) \in P_G} P(A \rightarrow \alpha)^{f(A \rightarrow \alpha; \psi)}$$

Statistical Parsing$_{17}$

P(t) -- Probability of a tree t (product of probabilities of the rules generating it.

P(w_{1n}) -- Probability of a sentence is the sum of the probabilities of all the valid parse trees of the sentence

$$P(w_{1n}) = \Sigma_j P(w_{1n}, t) \quad \text{where } t \text{ is a parse of } w_{1n}$$

$$= \Sigma_j P(t)$$

Statistical Parsing$_{18}$

- Positional invariance:

  The probability of a subtree is independent of its position in the derivation tree

- Context-free

  Independence from ancestors

Statistical Parsing$_{19}$

Parameter estimation

- Supervised learning
  - From a treebank
- Non supervised learning

Statistical Parsing$_{20}$

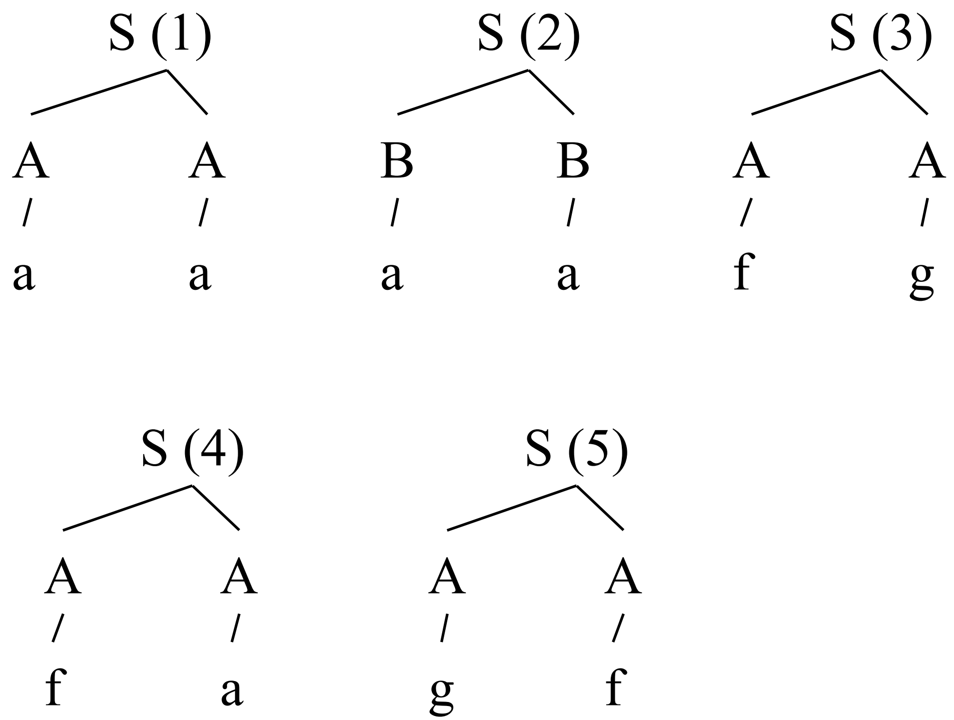
A Penn Treebank tree (POS tags not shown)

```
( (S (NP-SBJ The move)
     (VP followed
         (NP (NP a round)
             (PP of
                 (NP (NP similar increases)
                     (PP by
                         (NP other lenders))
                     (PP against
                         (NP Arizona real estate loans)))))
         ,
         (S-ADV (NP-SBJ *)
                (VP reflecting
                    (NP (NP a continuing decline)
                        (PP-LOC in
                                (NP that market))))))
     .))
```

374

Treebank grammars$_1$

Consider a treebank containing the following trees



Treebank grammars$_2$

Supose that (1) occurs 40 times, (2) occurs 10 times, (3) occurs 5 times, (4) occurs 5 times, and (5) occurs once.

We want to induce a SCFG reflexing this treebank.

Parameter estimation

$$\Sigma_j \mathrm{P}(N^i \rightarrow \zeta^j | N^i) = 1$$

Treebank grammars$_3$

Rules

S → A A	:	40 + 5 + 5 + 1 = 51
S → B B	:	10
A → *a*	:	40 + 40 + 5 = 85
A → *f*	:	5 + 5 + 1 = 11
A → *g*	:	5 + 1 = 6
B → *a*	:	10

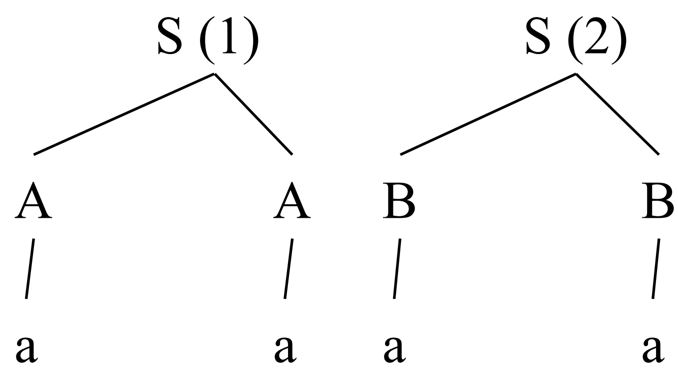
Treebank grammars$_4$

Parameters maximizing global likelihood of the corpus:

G	Frequency	Total	Probability
S → A A	51	61	0.836
S → B B	10	61	0.164
A → *a*	85	102	0.833
A → *f*	11	102	0.108
A → *g*	6	102	0.059
B → *a*	10	10	1.0

Treebank grammars$_5$

Given this parametrization,what is the most likely parse for "a a"?



$$P(1) = P(S \rightarrow A\ A) * P(A \rightarrow a) * P(A \rightarrow a)$$
$$= 0.836 * 0.833 * 0.833 = 0.580$$
$$P(2) = P(S \rightarrow B\ B) * P(B \rightarrow a) * P(B \rightarrow a)$$
$$= 0.164 * 1.0 * 1.0 = 0.164$$

Treebank grammars$_8$

-Removing non lnguistically valid rules

- Assign probabilities (MLE) to the initial rules

- Remove a rule except in the the probability of the structure built from its application is greater than the probability of building applicating simpler rules.

- Thresholding Removing rules occurring < n times

	Full	Simply thresholded	Fully compacted	Linguistically Compacted Grammar 1	Linguistically Compacted Grammar 2
Recall	70.55	70.78	30.93	71.55	70.76
precision	77.89	77.66	19.18	72.19	77.21
Grammar size	15,421	7,278	1,122	4,820	6,417