Treebanks

Generally

- A *treebank* is a corpus paired with 'gold-standard' (syntactico-semantic) analyses
- Created by manual annotation, typically with computational support (e.g. some automated processing plus correction)
- Can provide training data for machine learning (of parsers).

Treebanks

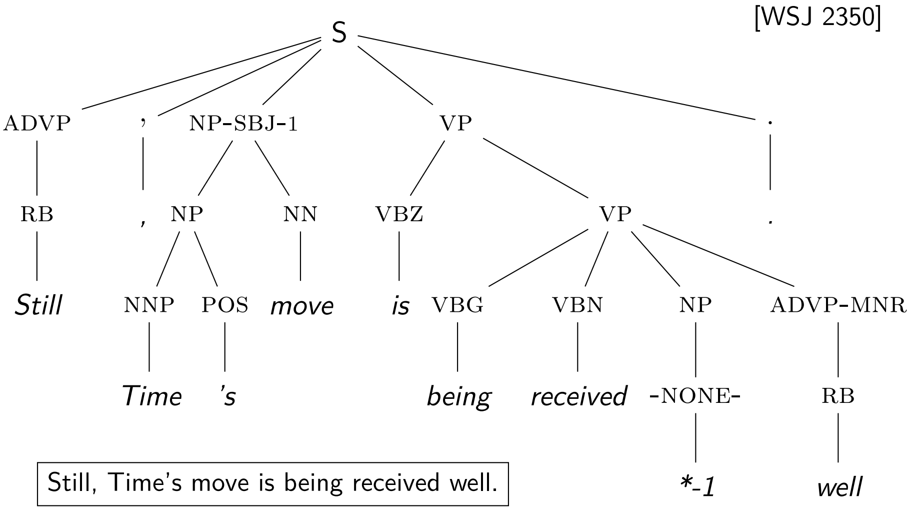
Generally

- A *treebank* is a corpus paired with 'gold-standard' (syntactico-semantic) analyses
- Created by manual annotation, typically with computational support (e.g. some automated processing plus correction)
- Can provide training data for machine learning (of parsers).

Penn Treebank (Marcus et al., 1993)

- About one million tokens of Wall Street Journal text
- Hand-corrected PoS annotation using 45 word classes
- Manual annotation with (somewhat) coarse constituent structure
- The 'mother' of all treebanks; still in wide use today.

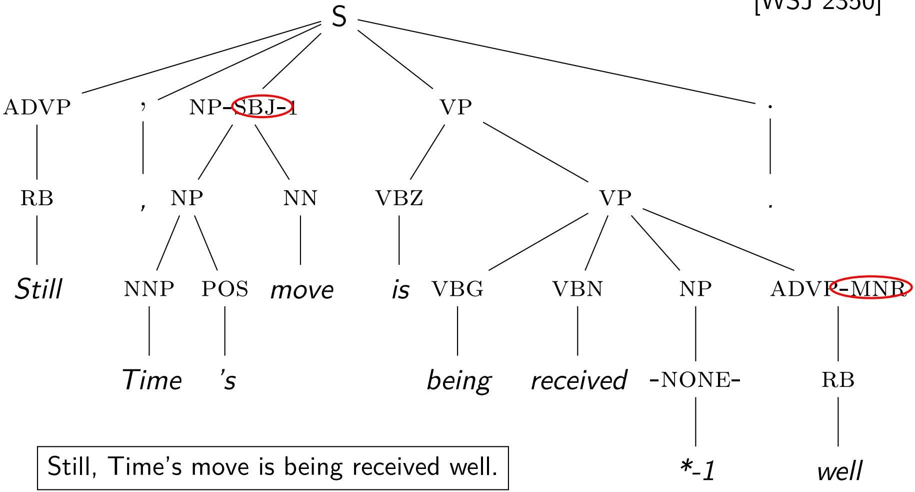
One Example from the Penn Treebank



Still, Time's move is being received well.

One Example from the Penn Treebank

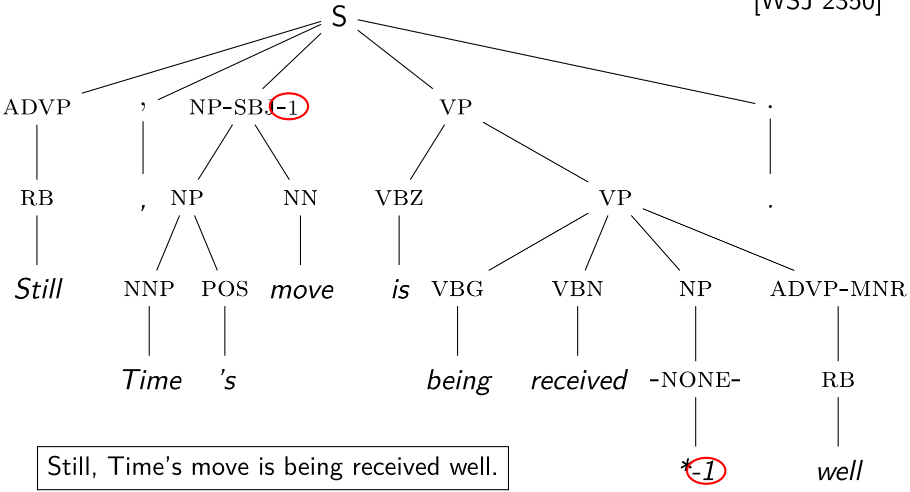
[WSJ 2350]



Still, Time's move is being received well.

One Example from the Penn Treebank

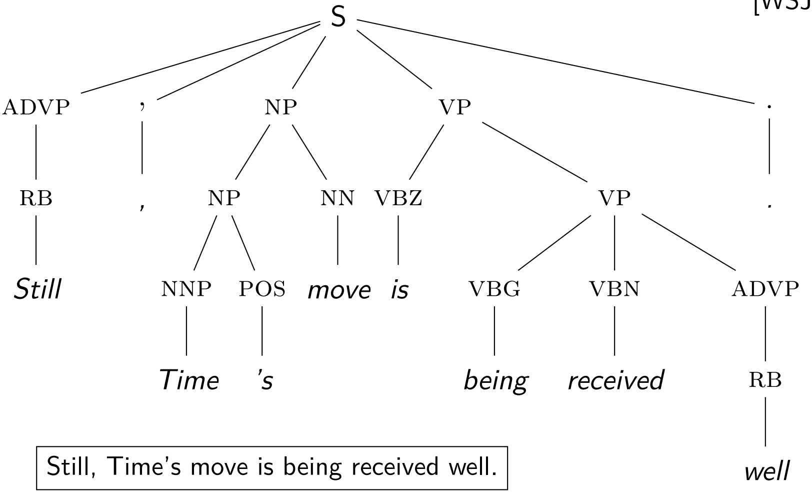
[WSJ 2350]



Still, Time's move is being received well.

Elimination of Traces and Functions

[WSJ 2350]



Still, Time's move is being received well.

Probabilistic Context-Free Grammars

- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.

Probabilistic Context-Free Grammars

- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.
- Probabilistic context-free grammars (PCFGs) augment CFGs by adding probabilities to each production, e.g.
  - S $\rightarrow$ NP VP 0.6
  - S $\rightarrow$ NP VP PP 0.4
- These are conditional probabilities: the probability of the right hand side (RHS), given the left hand side (LHS)
  - $P(S \rightarrow NP\ VP) = P(NP\ VP|S)$

Probabilistic Context-Free Grammars

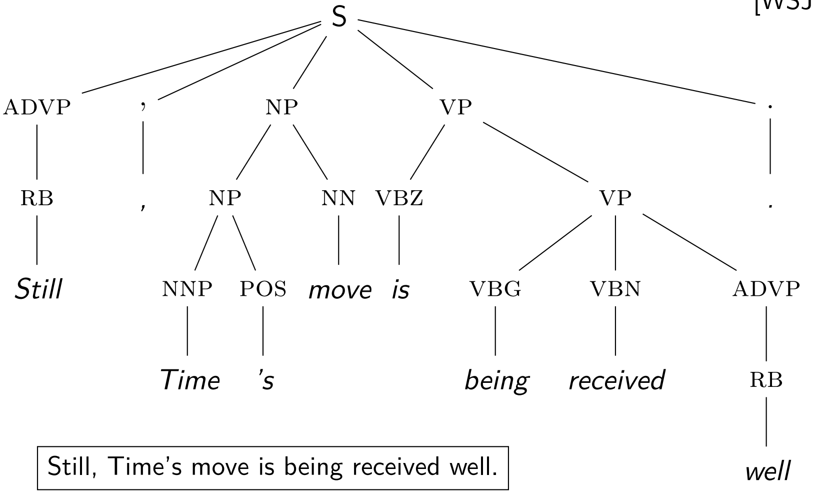
- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.
- Probabilistic context-free grammars (PCFGs) augment CFGs by adding probabilities to each production, e.g.
  - $S \rightarrow NP\ VP$ 0.6
  - $S \rightarrow NP\ VP\ PP$ 0.4
- These are conditional probabilities: the probability of the right hand side (RHS), given the left hand side (LHS)
  - $P(S \rightarrow NP\ VP) = P(NP\ VP|S)$
- The probability of a complete tree is the product of rule probabilities

Probabilistic Context-Free Grammars

- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.
- Probabilistic context-free grammars (PCFGs) augment CFGs by adding probabilities to each production, e.g.
  - S $\rightarrow$ NP VP 0.6
  - S $\rightarrow$ NP VP PP 0.4
- These are conditional probabilities: the probability of the right hand side (RHS), given the left hand side (LHS)
  - $P(S \rightarrow NP\ VP) = P(NP\ VP|S)$
- The probability of a complete tree is the product of rule probabilities
- We can learn these probabilities from a treebank, much like the estimation of HMM probabilities: Maximum Likelihood Estimation.

Estimating PCFGs (1/3)

[WSJ 2350]



Still, Time's move is being received well.

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

Estimating PCFGs (2/3)

RB $\rightarrow$ Still 1

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB $\rightarrow$ Still	1
ADVP $\rightarrow$ RB	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "’s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → ’s	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB $\rightarrow$ Still	1
ADVP $\rightarrow$ RB	1
, $\rightarrow$,	1
NNP $\rightarrow$ Time	1
POS $\rightarrow$'s	1
NP $\rightarrow$ NNP POS	1
NN $\rightarrow$ move	1
NP $\rightarrow$ NP NN	1
VBZ $\rightarrow$ is	1
VBG $\rightarrow$ being	1
VBN $\rightarrow$ received	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB $\to$ Still	1
ADVP $\to$ RB	1
, $\to$,	1
NNP $\to$ Time	1
POS $\to$'s	1
NP $\to$ NNP POS	1
NN $\to$ move	1
NP $\to$ NP NN	1
VBZ $\to$ is	1
VBG $\to$ being	1
VBN $\to$ received	1
RB $\to$ well	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "’s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → ’s	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB $\rightarrow$ Still	1
ADVP $\rightarrow$ RB	2
, $\rightarrow$,	1
NNP $\rightarrow$ Time	1
POS $\rightarrow$'s	1
NP $\rightarrow$ NNP POS	1
NN $\rightarrow$ move	1
NP $\rightarrow$ NP NN	1
VBZ $\rightarrow$ is	1
VBG $\rightarrow$ being	1
VBN $\rightarrow$ received	1
RB $\rightarrow$ well	1
VP $\rightarrow$ VBN ADVP	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1
. → .	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1
. → .	1
S → ADVP , NP VP .	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1
. → .	1
S → ADVP , NP VP .	1
START → S	1

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

S $\rightarrow$ ADVP , NP VP .	50	S $\rightarrow$ NP VP .	400
S $\rightarrow$ NP VP PP .	350	S $\rightarrow$ VP !	100
S $\rightarrow$ NP VP S .	200	S $\rightarrow$ NP VP	50

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

S $\rightarrow$ ADVP , NP VP .	50	S $\rightarrow$ NP VP .	400
S $\rightarrow$ NP VP PP .	350	S $\rightarrow$ VP !	100
S $\rightarrow$ NP VP S .	200	S $\rightarrow$ NP VP	50

$$P(S \rightarrow ADVP\ ,\ NP\ VP\ .) \approx \frac{C(S \rightarrow ADVP\ ,\ NP\ VP\ .)}{C(S)}$$

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

S → ADVP , NP VP .	50	S → NP VP .	400
S → NP VP PP .	350	S → VP !	100
S → NP VP S .	200	S → NP VP	50

$$
\begin{aligned}
P(S \rightarrow ADVP\ ,\ NP\ VP\ .) &\approx \frac{C(S \rightarrow ADVP\ ,\ NP\ VP\ .)}{C(S)} \\
&= \frac{50}{1150} \\
&= 0.0435
\end{aligned}
$$

Viterbi Decoding over the Parse Forest

- Recall the Viterbi algorithm for HMMs

$$v_i(s) = \max_{k=1}^{L} \left[v_{i-1}(k) \cdot P(s|k) \cdot P(o_i|s) \right]$$

Viterbi Decoding over the Parse Forest

- Recall the Viterbi algorithm for HMMs

$$v_i(s) = \max_{k=1}^{L} \left[v_{i-1}(k) \cdot P(s|k) \cdot P(o_i|s) \right]$$

- Over the (result edges from the) parse forest, compute Viterbi scores for sub-trees of increasing size:

$$v(\alpha) = \max \left[P(\beta_1, \ldots \beta_n|\alpha) \times \prod_{i=1}^{n} v(\beta_i) \right]$$

Viterbi Decoding over the Parse Forest

- Recall the Viterbi algorithm for HMMs

$$v_i(s) = \max_{k=1}^{L} [v_{i-1}(k) \cdot P(s|k) \cdot P(o_i|s)]$$

- Over the (result edges from the) parse forest, compute Viterbi scores for sub-trees of increasing size:

$$v(\alpha) = \max \left[P(\beta_1, \ldots \beta_n|\alpha) \times \prod_{i=1}^{n} v(\beta_i) \right]$$

- Similar to HMM decoding, we also need to keep track of the set of daughters that led to the maximum probability.

Exercise (1): Natural Language Ambiguity

Assume the following 'toy' grammar of English:

S → NP
NP → Det N
N → N N

Det → *the*
N → *kitchen* | *gold* | *towel* | *rack*

Exercise (1): Natural Language Ambiguity

Assume the following 'toy' grammar of English:

$$S \rightarrow NP$$
$$NP \rightarrow Det\ N$$
$$N \rightarrow N\ N$$

$$Det \rightarrow \textit{the}$$
$$N \rightarrow \textit{kitchen} \mid \textit{gold} \mid \textit{towel} \mid \textit{rack}$$

(1) How many different syntactic analyses, if any, does the grammar assign to the following strings?

(a) *the kitchen towel rack*
(b) *the kitchen gold towel rack*

Exercise (2): CKY Parsing

Assume the following grammar and CKY parse table:

S → NP VP
VP → V NP
VP → VP PP
NP → NP VP
PP → P NP

	1	2	3	4	5
0	NP		S		S
1		V	VP		VP
2			NP		NP
3				P	PP
4					NP

Exercise (2): CKY Parsing

Assume the following grammar and CKY parse table:

S → NP VP
VP → V NP
VP → VP PP
NP → NP VP
PP → P NP

	1	2	3	4	5
0	NP		S		S
1		V	VP		VP
2			NP		NP
3				P	PP
4					NP

(2) Which pair(s) of 'input' cells and which production(s) give rise to the derivation of category S in 'target' cell $\langle 0, 5 \rangle$?

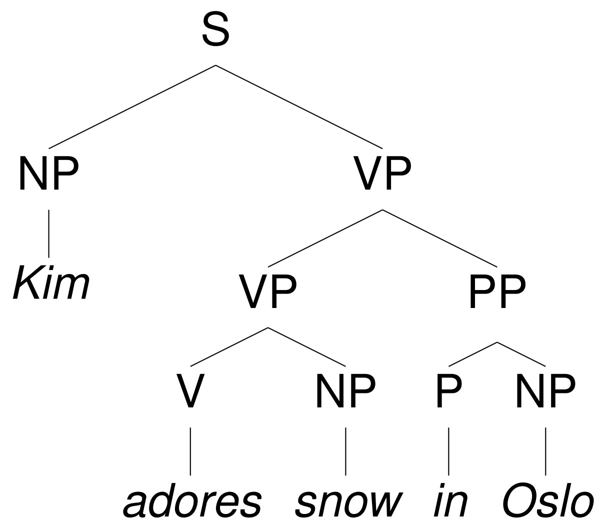
Outlook

After the Easter Break

- Dependency syntax
- Transition-based dependency parsing
- Using syntactic structure

Parsing with CFGs: Moving to a Procedural View

S → NP VP
VP → V | V NP | VP PP
NP → NP PP
PP → P NP
NP → Kim | snow | Oslo
V → adores
P → in



All Complete Derivations

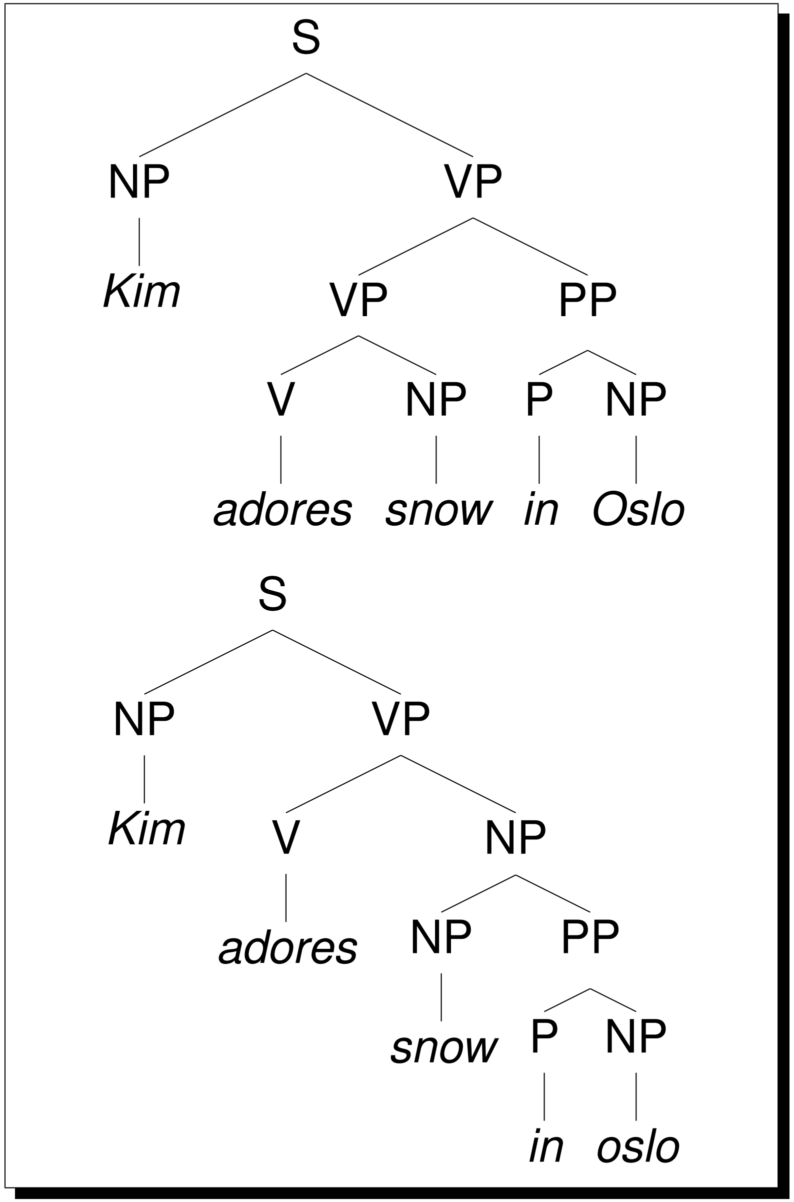
- are rooted in the start symbol S;
- label internal nodes with categories $\in C$, leafs with words $\in \Sigma$;
- instantiate a grammar rule $\in P$ at each local subtree of depth one.

Parsing with CFGs: Moving to a Procedural View

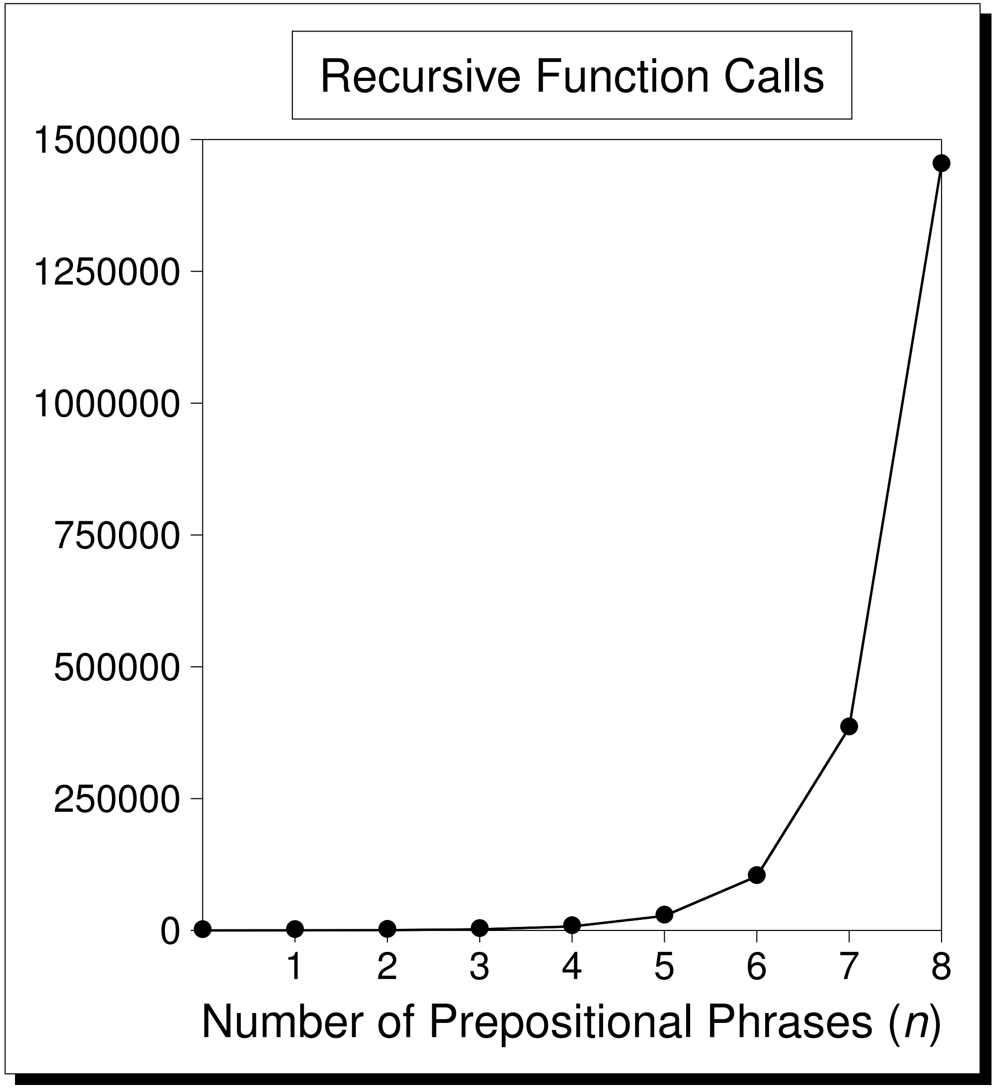
S $\rightarrow$ NP VP
VP $\rightarrow$ V | V NP | VP PP
NP $\rightarrow$ NP PP
PP $\rightarrow$ P NP
NP $\rightarrow$ Kim | snow | Oslo
V $\rightarrow$ adores
P $\rightarrow$ in

All Complete Derivations

- are rooted in the start symbol S;
- label internal nodes with categories $\in C$, leafs with words $\in \Sigma$;
- instantiate a grammar rule $\in P$ at each local subtree of depth one.



Quantifying the Complexity of the Parsing Task

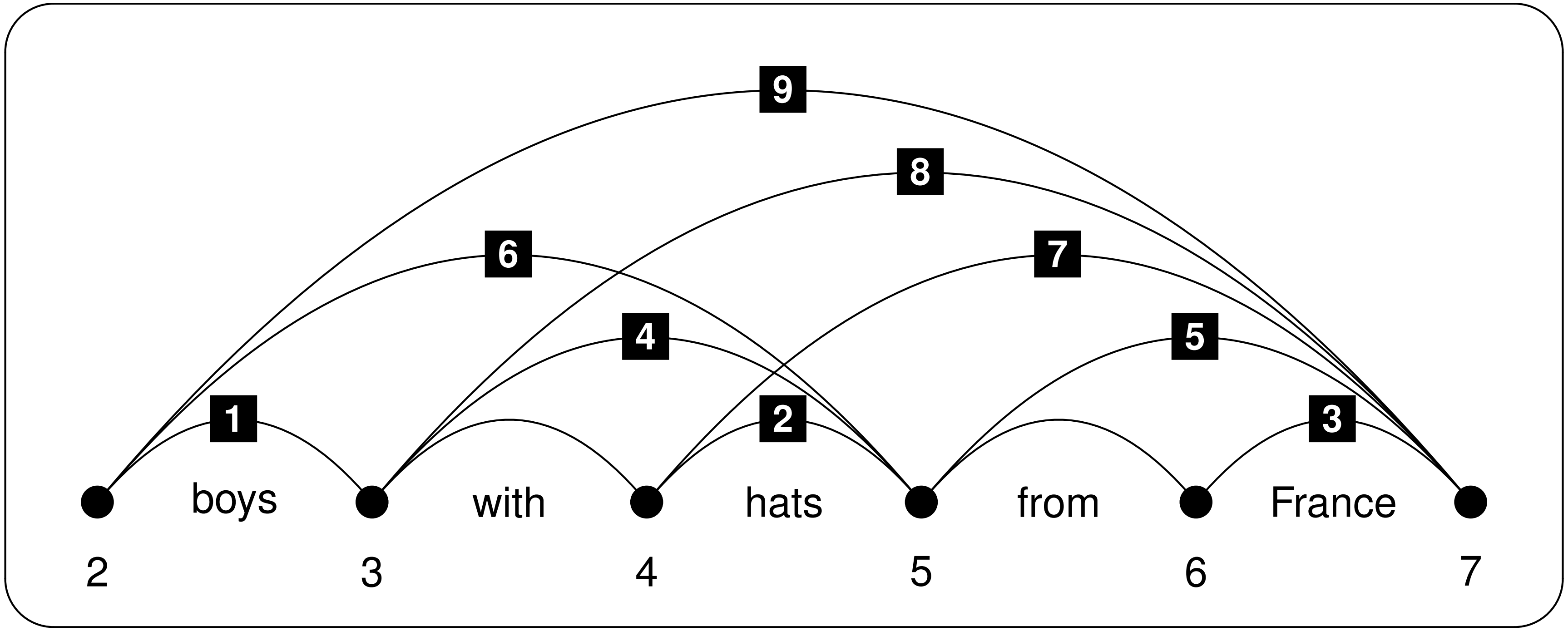


Kim adores snow (in Oslo)^n

n	trees	calls
0	1	46
1	2	170
2	5	593
3	14	2,093
4	42	7,539
5	132	27,627
6	429	102,570
7	1430	384,566
8	4862	1,452,776
⋮	⋮	⋮

A Key Insight: Local Ambiguity

- For many substrings, more than one way of deriving the same category;
- NPs: **1** | **2** | **3** | **6** | **7** | **9**; PPs: **4** | **5** | **8**; **9** $\equiv$ **1** + **8** | **6** + **5**;
- *parse forest* — a single item represents multiple trees [Billot & Lang, 89].



The CKY (Cocke, Kasami, & Younger) Algorithm

Kim adored snow in Oslo

	1	2	3	4	5
0	NP		S		S
1		V	VP		VP
2			NP		NP
3				P	PP
4					NP

The CKY (Cocke, Kasami, & Younger) Algorithm

Kim adored snow in Oslo

	1	2	3	4	5
0	NP		S		S
1		V	VP		VP
2			NP		NP
3				P	PP
4					NP

The CKY (Cocke, Kasami, & Younger) Algorithm

for $(0 \leq i < |\textit{input}|)$ do
  $\textit{chart}_{[i,i+1]} \leftarrow \{\alpha \,|\, \alpha \rightarrow \textit{input}_i \in P\}$;
for $(1 \leq l < |\textit{input}|)$ do
  for $(0 \leq i < |\textit{input}| - l)$ do
    for $(1 \leq j \leq l)$ do
      if $(\alpha \rightarrow \beta_1\,\beta_2 \in P \wedge \beta_1 \in \textit{chart}_{[i,i+j]} \wedge \beta_2 \in \textit{chart}_{[i+j,i+l+1]})$ then
        $\textit{chart}_{[i,i+l+1]} \leftarrow \textit{chart}_{[i,i+l+1]} \cup \{\alpha\}$;

Kim adored snow in Oslo

	1	2	3	4	5
0	NP		S		S
1		V	VP		VP
2			NP		NP
3				P	PP
4					NP

Chart Parsing — Specialized Dynamic Programming

Basic Notions

- Use *chart* to record partial analyses, indexing them by string positions;
- count inter-word vertices; CKY: chart row is *start*, column *end* vertex;
- treat multiple ways of deriving the same category for some substring as *equivalent*; pursue only once when combining with other constituents.

Chart Parsing — Specialized Dynamic Programming

Basic Notions

- Use *chart* to record partial analyses, indexing them by string positions;
- count inter-word vertices; CKY: chart row is *start*, column *end* vertex;
- treat multiple ways of deriving the same category for some substring as *equivalent*; pursue only once when combining with other constituents.

Key Benefits

- Dynamic programming (memoization): avoid recomputation of results;
- efficient indexing of constituents: no search by start or end positions;
- compute *parse forest* with exponential 'extension' in *polynomial* time.

In Conclusion—What Happened this Far

Syntactic Structure

- Languages (formal or natural) exhibit complex, hierarchical structures;
- grammars encode rules of the language: dominance and sequencing;
- context-free grammar 'generates' a language: strings and derivations;
- ambiguity in natural language grows exponentially: a search problem;
- bounding (or 'packing') of local ambiguity is mandatory for tractability;
- chart parsing uses dynamic programming: free order of computation.

In Conclusion—What Happened this Far

Syntactic Structure

- Languages (formal or natural) exhibit complex, hierarchical structures;
- grammars encode rules of the language: dominance and sequencing;
- context-free grammar 'generates' a language: strings and derivations;
- ambiguity in natural language grows exponentially: a search problem;
- bounding (or 'packing') of local ambiguity is mandatory for tractability;
- chart parsing uses dynamic programming: free order of computation.

Coming up Next

- Treebank parsing; Viterbi adaptation on parse forest; parser evaluation.

Ambiguity Resolution is a (Major) Challenge

The Problem

- Even moderately complex sentences often have (very) *many* analyses;
- in most applications, computing all possible readings is hardly helpful;
- identifying the 'correct' (intended) analysis is an 'AI-complete' problem.

Ambiguity Resolution is a (Major) Challenge

The Problem

- Even moderately complex sentences often have (very) *many* analyses;
- in most applications, computing all possible readings is hardly helpful;
- identifying the 'correct' (intended) analysis is an 'AI-complete' problem.

Once Again: Probabilities to the Rescue

- Design and use statistical models to select among competing analyses;
- for string S, some analyses T_i are more or less likely: maximize $P(T_i|S)$;

$\rightarrow$ Probabilistic Context Free Grammar (PCFG) is a CFG plus probabilities.

Treebanks

Generally

- A *treebank* is a corpus paired with 'gold-standard' (syntactico-semantic) analyses
- Created by manual annotation, typically with computational support (e.g. some automated processing plus correction)
- Can provide training data for machine learning (of parsers).

Treebanks

Generally

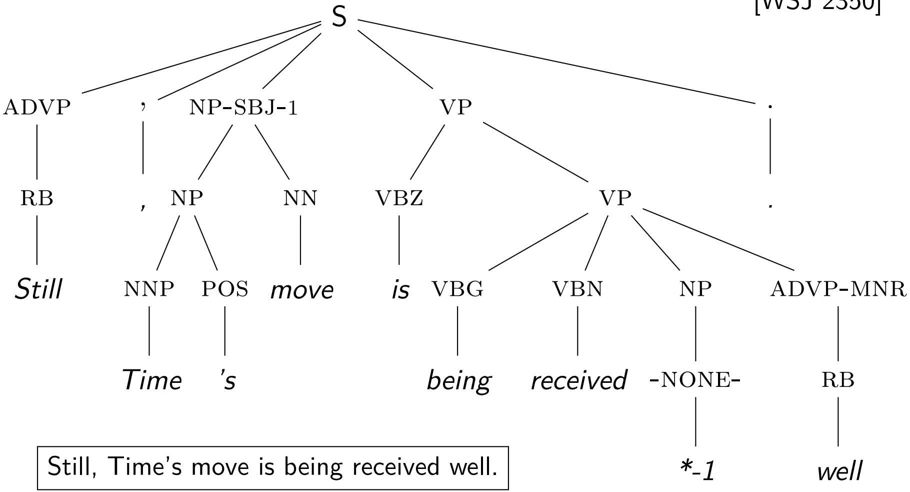
- A *treebank* is a corpus paired with 'gold-standard' (syntactico-semantic) analyses
- Created by manual annotation, typically with computational support (e.g. some automated processing plus correction)
- Can provide training data for machine learning (of parsers).

Penn Treebank (Marcus et al., 1993)

- About one million tokens of Wall Street Journal text
- Hand-corrected PoS annotation using 45 word classes
- Manual annotation with (somewhat) coarse constituent structure
- The 'mother' of all treebanks; still in wide use today.

One Example from the Penn Treebank

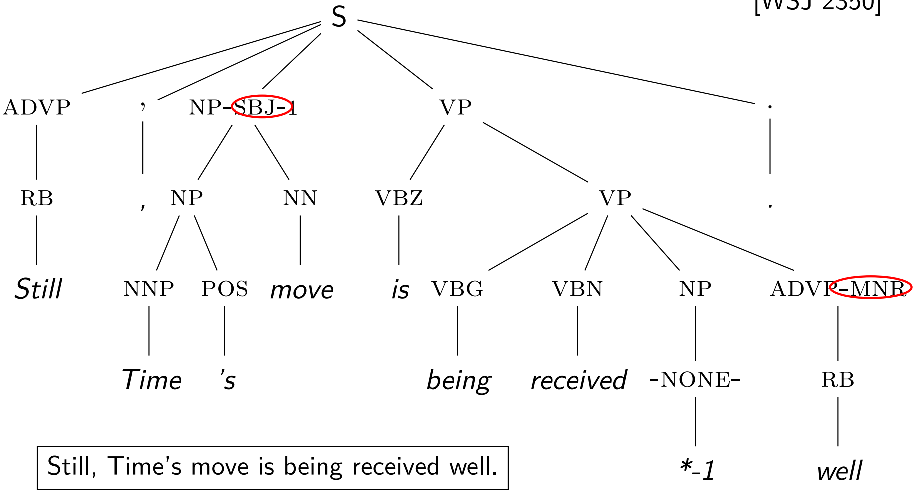
[WSJ 2350]



Still, Time's move is being received well.

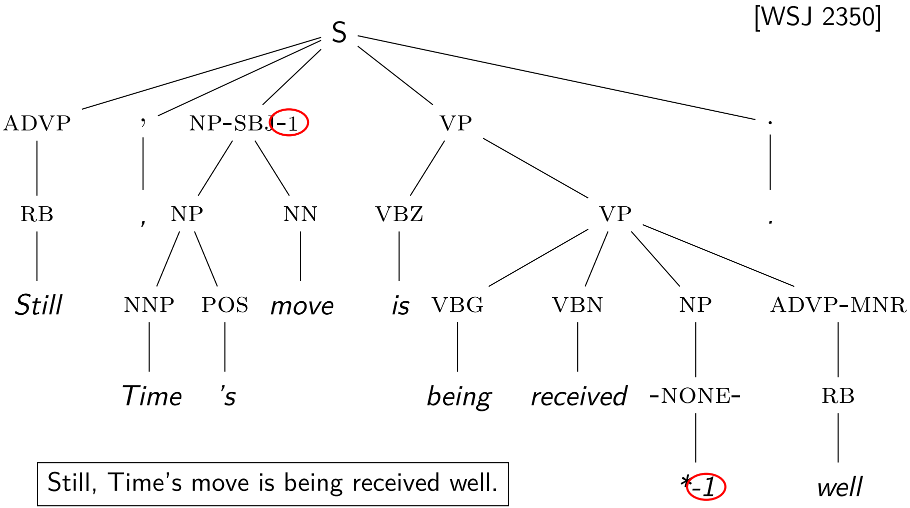
One Example from the Penn Treebank

[WSJ 2350]



Still, Time's move is being received well.

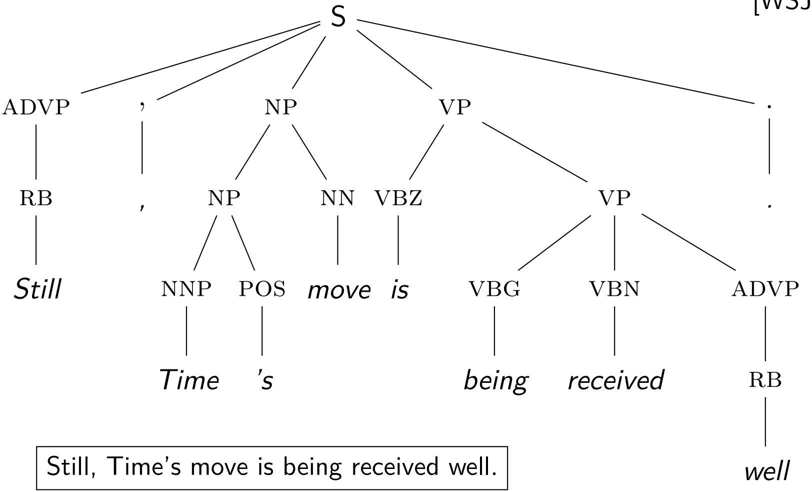
One Example from the Penn Treebank



Still, Time's move is being received well.

Elimination of Traces and Functions

[WSJ 2350]



Still, Time's move is being received well.

Probabilistic Context-Free Grammars

- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.

Probabilistic Context-Free Grammars

- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.
- Probabilistic context-free grammars (PCFGs) augment CFGs by adding probabilities to each production, e.g.
  - S $\rightarrow$ NP VP 0.6
  - S $\rightarrow$ NP VP PP 0.4
- These are conditional probabilities: the probability of the right hand side (RHS), given the left hand side (LHS)
  - $P(S \rightarrow NP\ VP) = P(NP\ VP|S)$

Probabilistic Context-Free Grammars

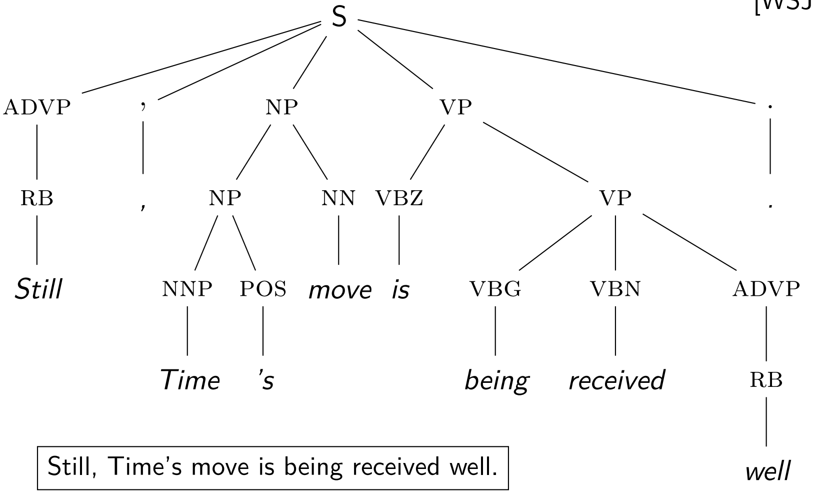
- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.
- Probabilistic context-free grammars (PCFGs) augment CFGs by adding probabilities to each production, e.g.
  - $S \rightarrow NP\ VP$ 0.6
  - $S \rightarrow NP\ VP\ PP$ 0.4
- These are conditional probabilities: the probability of the right hand side (RHS), given the left hand side (LHS)
  - $P(S \rightarrow NP\ VP) = P(NP\ VP|S)$
- The probability of a complete tree is the product of rule probabilities

Probabilistic Context-Free Grammars

- Towards statistical parsing: Not just interested in which trees can apply to a sentence, but also which tree is most likely.
- Probabilistic context-free grammars (PCFGs) augment CFGs by adding probabilities to each production, e.g.
  - S $\rightarrow$ NP VP 0.6
  - S $\rightarrow$ NP VP PP 0.4
- These are conditional probabilities: the probability of the right hand side (RHS), given the left hand side (LHS)
  - P(S $\rightarrow$ NP VP) $=$ P(NP VP$|$S)
- The probability of a complete tree is the product of rule probabilities
- We can learn these probabilities from a treebank, much like the estimation of HMM probabilities: Maximum Likelihood Estimation.

Estimating PCFGs (1/3)

[WSJ 2350]



Still, Time's move is being received well.

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still 1

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

Estimating PCFGs (2/3)

RB → Still	1
ADVP → RB	1

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB $\rightarrow$ Still	1
ADVP $\rightarrow$ RB	1
, $\rightarrow$,	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB $\rightarrow$ Still	1
ADVP $\rightarrow$ RB	1
, $\rightarrow$,	1
NNP $\rightarrow$ Time	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB $\rightarrow$ Still	1
ADVP $\rightarrow$ RB	1
, $\rightarrow$,	1
NNP $\rightarrow$ Time	1
POS $\rightarrow$'s	1
NP $\rightarrow$ NNP POS	1
NN $\rightarrow$ move	1
NP $\rightarrow$ NP NN	1
VBZ $\rightarrow$ is	1
VBG $\rightarrow$ being	1
VBN $\rightarrow$ received	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	1
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1
. → .	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1
. → .	1
S → ADVP , NP VP .	1

Estimating PCFGs (2/3)

```
(S
   (ADVP (RB "Still"))
   (, ",")
   (NP
      (NP (NNP "Time") (POS "'s"))
      (NN "move"))
   (VP
      (VBZ "is")
      (VP
         (VBG "being")
         (VP
           (VBN "received")
           (ADVP (RB "well")))))
   (. "."))
```

RB → Still	1
ADVP → RB	2
, → ,	1
NNP → Time	1
POS → 's	1
NP → NNP POS	1
NN → move	1
NP → NP NN	1
VBZ → is	1
VBG → being	1
VBN → received	1
RB → well	1
VP → VBN ADVP	1
VP → VBG VP	1
. → .	1
S → ADVP , NP VP .	1
START → S	1

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

S → ADVP , NP VP .	50	S → NP VP .	400
S → NP VP PP .	350	S → VP !	100
S → NP VP S .	200	S → NP VP	50

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

S → ADVP , NP VP .	50	S → NP VP .	400
S → NP VP PP .	350	S → VP !	100
S → NP VP S .	200	S → NP VP	50

$$P(S \rightarrow ADVP\ ,\ NP\ VP\ .) \approx \frac{C(S \rightarrow ADVP\ ,\ NP\ VP\ .)}{C(S)}$$

Estimating PCFGs (3/3)

Once we have counts of all the rules, we turn them into probabilities.

S → ADVP , NP VP .	50	S → NP VP .	400
S → NP VP PP .	350	S → VP !	100
S → NP VP S .	200	S → NP VP	50

$$P(S \rightarrow ADVP\ ,\ NP\ VP\ .) \approx \frac{C(S \rightarrow ADVP\ ,\ NP\ VP\ .)}{C(S)} = \frac{50}{1150} = 0.0435$$

Viterbi Decoding over the Parse Forest

- Recall the Viterbi algorithm for HMMs

$$v_i(s) = \max_{k=1}^{L} \left[v_{i-1}(k) \cdot P(s|k) \cdot P(o_i|s) \right]$$

Viterbi Decoding over the Parse Forest

- Recall the Viterbi algorithm for HMMs

$$v_i(s) = \max_{k=1}^{L} \left[v_{i-1}(k) \cdot P(s|k) \cdot P(o_i|s) \right]$$

- Over the (result edges from the) parse forest, compute Viterbi scores for sub-trees of increasing size:

$$v(\alpha) = \max \left[P(\beta_1, \ldots \beta_n | \alpha) \times \prod_{i=1}^{n} v(\beta_i) \right]$$

Viterbi Decoding over the Parse Forest

- Recall the Viterbi algorithm for HMMs

$$v_i(s) = \max_{k=1}^{L} \left[v_{i-1}(k) \cdot P(s|k) \cdot P(o_i|s) \right]$$

- Over the (result edges from the) parse forest, compute Viterbi scores for sub-trees of increasing size:

$$v(\alpha) = \max \left[P(\beta_1, \ldots \beta_n|\alpha) \times \prod_{i=1}^{n} v(\beta_i) \right]$$

- Similar to HMM decoding, we also need to keep track of the set of daughters that led to the maximum probability.

Exercise (1): Natural Language Ambiguity

Assume the following 'toy' grammar of English:

S → NP
NP → Det N
N → N N

Det → *the*
N → *kitchen* | *gold* | *towel* | *rack*

Exercise (1): Natural Language Ambiguity

Assume the following 'toy' grammar of English:

S → NP
NP → Det N
N → N N

Det → *the*
N → *kitchen* | *gold* | *towel* | *rack*

(1) How many different syntactic analyses, if any, does the grammar assign to the following strings?

(a) *the kitchen towel rack*
(b) *the kitchen gold towel rack*

Exercise (2): CKY Parsing

Assume the following grammar and CKY parse table:

S → NP VP
VP → V NP
VP → VP PP
NP → NP VP
PP → P NP

	1	2	3	4	5
0	NP		S		S
1		V	VP		VP
2			NP		NP
3				P	PP
4					NP

Exercise (2): CKY Parsing

Assume the following grammar and CKY parse table:

S → NP VP
VP → V NP
VP → VP PP
NP → NP VP
PP → P NP

	1	2	3	4	5
0	NP		S		S
1		V	VP		VP
2			NP		NP
3				P	PP
4					NP

(2) Which pair(s) of ‘input’ cells and which production(s) give rise to the derivation of category S in ‘target’ cell $\langle 0, 5 \rangle$?

Outlook

After the Easter Break

- Dependency syntax
- Transition-based dependency parsing
- Using syntactic structure