Features of Statistical Parsers

Mark Johnson

Brown University

April, 2005

Joint work with Eugene Charniak, Michael Collins and students in BLLIP
Supported by NSF grants LIS 9720368 and IIS0095940

Talk outline

- Who cares about parsing?
- PCFGs, maximum likelihood estimation and (lack of) independence
- Dependence, arbitrary features and exponential parsing models
- Joint vs. conditional maximum likelihood estimation of exponential models
- Features of an exponential parsing model

Why parsing?

- Applications
  - Recovering syntactic structure
    - Information extraction
    - Question answering
    - Machine translation
  - Language modeling (distinguishing likely from unlikely sentences)
    - Speech recognition
    - Speech error detection and correction
- Science
  - First step on the path to meaning

Short history of parsing

- (1960s–1970s) basic dynamic programming algorithms, ATNs
  - inspired by linguistic theory (transformational grammar)
- (1980s–1990s) "unification" grammar (no deep structure)
  - parsers provably implement a specific linguistic grammar
  - grammar ~ axioms, UG ~ logic, parsing ~ deduction
  - emphasis on grammar formalisms
- (1980s–??) direct statistical models (e.g., n-gram models)
  - revolutionized speech recognition, practical applications
- (1990s–??) statistical models of linguistic structure
  - "curse of dimensionality" limits direct statistical models
  - emphasis on frequent constructions

N-gram models of strings

- Model dependencies between n adjacent words (*n-gram models*)
  - predict a word based on the $n-1$ preceding words

$$\$ \rightarrow the \rightarrow man \rightarrow in \rightarrow the \rightarrow hat \rightarrow drinks \rightarrow red \rightarrow wine \rightarrow \$$$

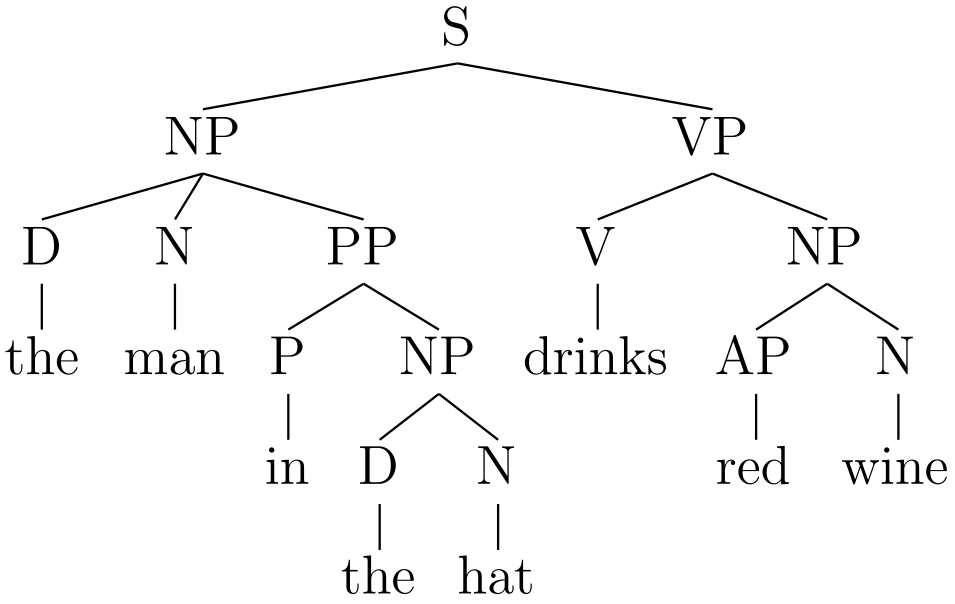
$$n = 1 : \mathrm{P}(hat);\ n = 2 : \mathrm{P}(hat|the);\ n = 3 : \mathrm{P}(hat|in, the)$$

- N-gram models are *universal approximators* as $n \rightarrow \infty$
- Probabilities estimated from real *corpora*
- Probability distinguishes "good" from "bad" sentences
- These simple models work surprisingly well because *they are lexicalized* and *most dependencies are local*

Limitations of n-gram models

- *Curse of dimensionality:* the number of possible n-grams (sequences of n words) grows exponentially with n
  - $n = 1 \Rightarrow V^n \approx 1,000$
  - $n = 2 \Rightarrow V^n \approx 1,000,000$
  - $n = 3 \Rightarrow V^n \approx 1,000,000,000$
- *Bias-variance dilemma:* As n gets larger ...
  - the model becomes more accurate (lower bias)
  - more data is needed to learn the model (higher variance)
- *Structured models* collect statistics on linguistic structures
  - only model linguistic dependencies (strong bias)
  - ignore other dependencies (lower variance)

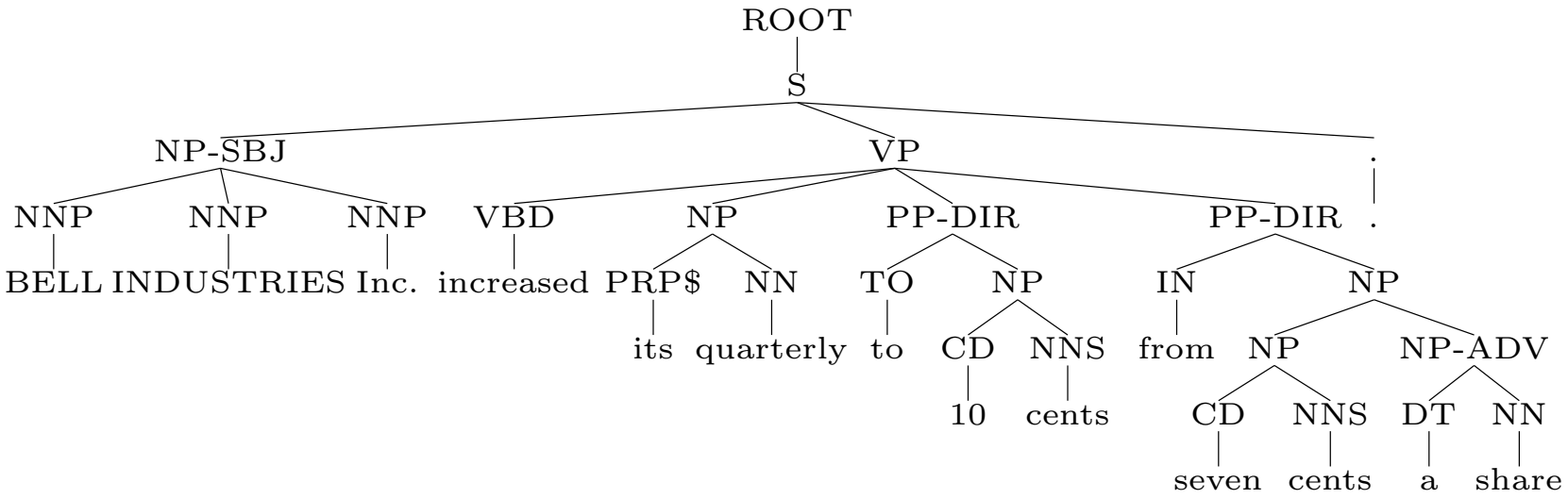
Probabilistic Context Free Grammars



$$
\begin{aligned}
\mathrm{P(t)} \quad = \quad & \mathrm{P(S \rightarrow NP\ VP)} \times \\
& \mathrm{P(NP \rightarrow D\ N\ PP)} \times \\
& \mathrm{P(D \rightarrow the)} \times \\
& \mathrm{P(N \rightarrow man)} \times \\
& \ldots
\end{aligned}
$$

- Rules are associated with *probabilities*
- Tree probability is the *product of rule probabilities*
- *Most probable tree* is "best guess" at correct syntactic structure

Treebank corpora



- The Penn treebank contains hand-annotated parse trees for $\sim 50,000$ sentences
- Treebanks also exist for the Brown corpus, the Switchboard corpus (spontaneous telephone conversations) and Chinese and Arabic corpora

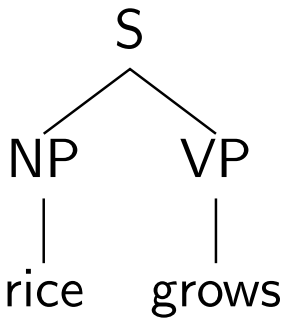
Estimating a grammar from a treebank

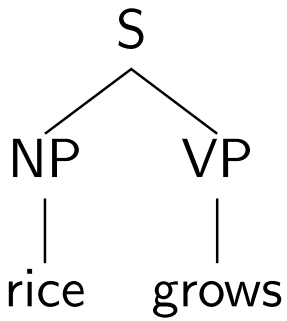
- *Maximum likelihood principle*: Choose the grammar and rule probabilities that make the trees in the corpus *as likely as possible*
  - read the rules off the trees
  - for PCFGs, set rule probabilities to the *relative frequency* of each rule in the treebank

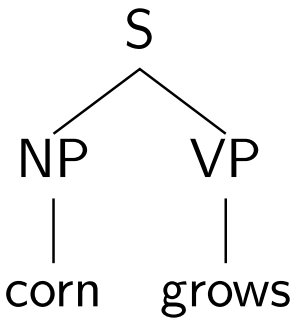
$$\mathrm{P}(\mathrm{VP} \rightarrow \mathrm{V\ NP}) = \frac{\text{Number of times VP} \rightarrow \text{V NP occurs}}{\text{Number of times VP occurs}}$$

- *If the language is generated by a PCFG and the treebank trees are its derivation trees, the estimated grammar converges to the true grammar.*

Estimating PCFGs from visible data





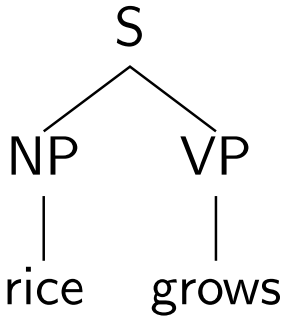


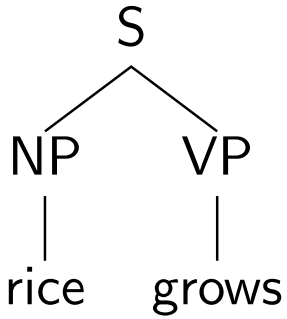
Rule	**Count**	**Rel Freq**
S $\rightarrow$ NP VP	3	1
NP $\rightarrow$ rice	2	2/3
NP $\rightarrow$ corn	1	1/3
VP $\rightarrow$ grows	3	1

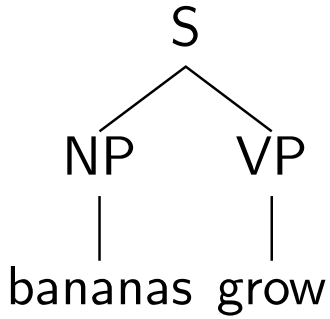
$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{rice}\quad\text{grows}\end{array}\right) = 2/3$$

$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{corn}\quad\text{grows}\end{array}\right) = 1/3$$

Non-local constraints and PCFG MLE







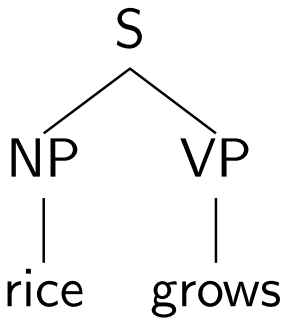
Rule	**Count**	**Rel Freq**
S → NP VP	3	1
NP → rice	2	2/3
NP → bananas	1	1/3
VP → grows	2	2/3
VP → grow	1	1/3

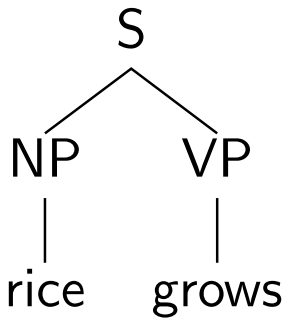
$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{rice}\quad\text{grows}\end{array}\right) = 4/9$$

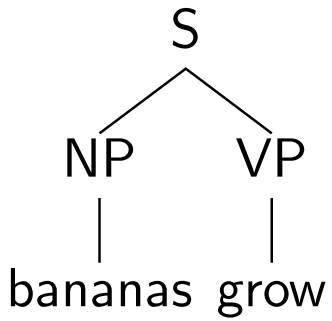
$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{bananas}\quad\text{grow}\end{array}\right) = 1/9$$

$$\textit{partition function } Z = 5/9$$

Dividing by partition function Z







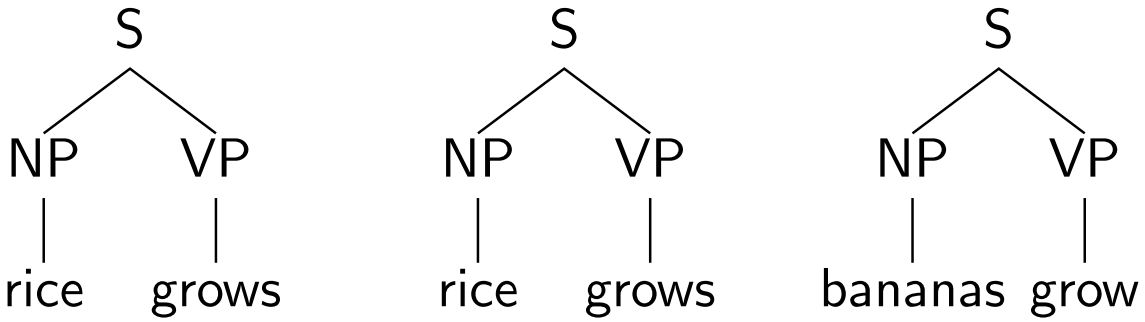
Rule	**Count**	**Rel Freq**
S $\rightarrow$ NP VP	3	1
NP $\rightarrow$ rice	2	2/3
NP $\rightarrow$ bananas	1	1/3
VP $\rightarrow$ grows	2	2/3
VP $\rightarrow$ grow	1	1/3

$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{rice}\quad\text{grows}\end{array}\right) = 4/9 \quad 4/5$$

$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{bananas}\quad\text{grow}\end{array}\right) = 1/9 \quad 1/5$$

$$Z = 5/9$$

Other values do better!



Rule	**Count**	**Rel Freq**
S → NP VP	3	1
NP → rice	2	2/3
NP → bananas	1	1/3
VP → grows	2	1/2
VP → grow	1	1/2

(Abney 1997)

$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{rice}\quad\text{grows}\end{array}\right) = 2/6 \quad 2/3$$

$$P\left(\begin{array}{c}\text{S}\\ \text{NP}\quad\text{VP}\\ \text{bananas}\quad\text{grow}\end{array}\right) = 1/6 \quad 1/3$$

$$Z = 3/6$$

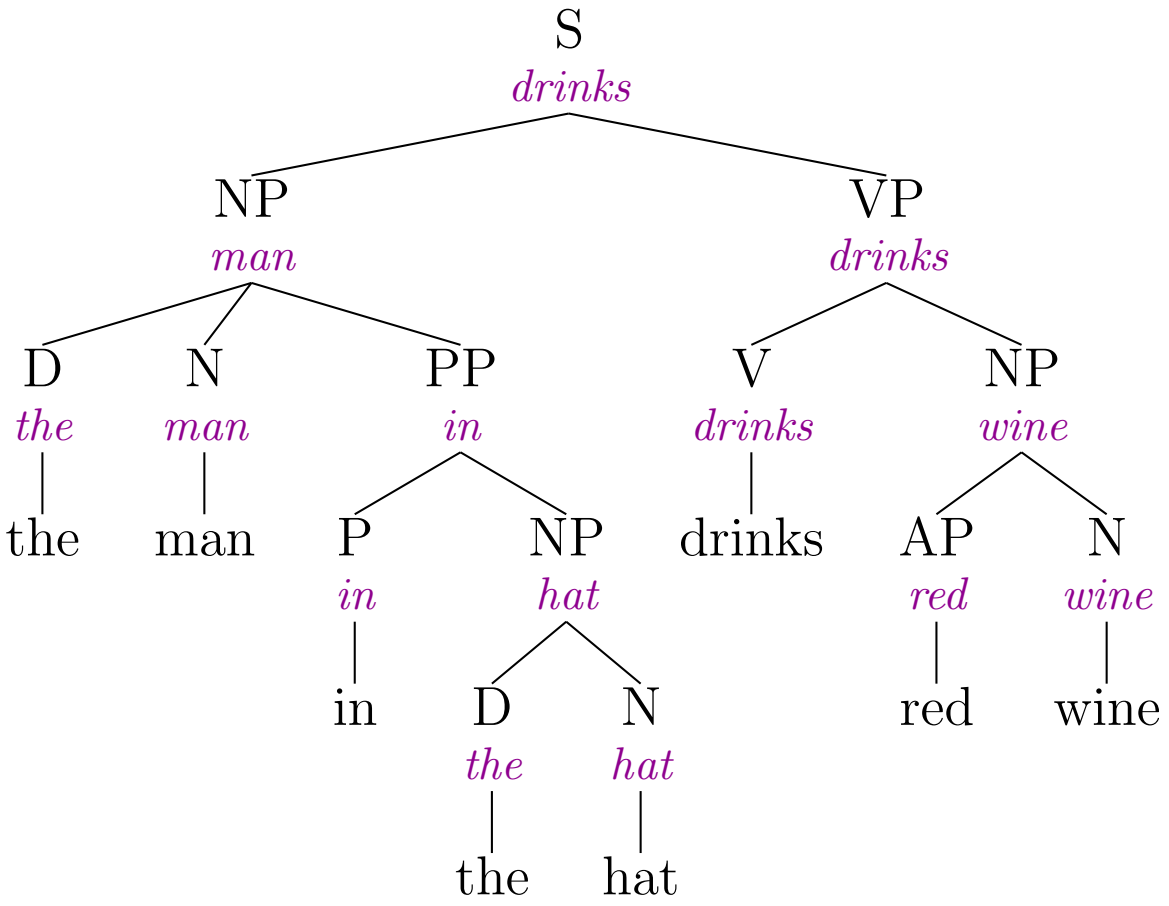
Make dependencies local – GPSG-style

rule	count	rel freq
$S \rightarrow \underset{+\text{singular}}{NP}\ \underset{+\text{singular}}{VP}$	2	2/3
$S \rightarrow \underset{+\text{plural}}{NP}\ \underset{+\text{plural}}{VP}$	1	1/3
$\underset{+\text{singular}}{NP} \rightarrow$ rice	2	1
$\underset{+\text{plural}}{NP} \rightarrow$ bananas	1	1
$\underset{+\text{singular}}{VP} \rightarrow$ grows	2	1
$\underset{+\text{plural}}{VP} \rightarrow$ grow	1	1

$$P\left(\begin{array}{c} S \\ \diagup\ \diagdown \\ \underset{+\text{singular}}{NP} \quad \underset{+\text{singular}}{VP} \\ | \qquad | \\ \text{rice} \quad \text{grows} \end{array}\right) = 2/3$$

$$P\left(\begin{array}{c} S \\ \diagup\ \diagdown \\ \underset{+\text{plural}}{NP} \quad \underset{+\text{plural}}{VP} \\ | \qquad | \\ \text{bananas} \quad \text{grow} \end{array}\right) = 1/3$$

"Head to head" dependencies



Rules:

$$\underset{drinks}{S} \rightarrow \underset{man}{NP}\ \underset{drinks}{VP}$$

$$\underset{drinks}{VP} \rightarrow \underset{drinks}{V}\ \underset{wine}{NP}$$

$$\underset{wine}{NP} \rightarrow \underset{red}{AP}\ \underset{wine}{N}$$

. . .

- *Lexicalization* captures syntactic and semantic dependencies
- Lexicalized structural preferences may be most important

Summary so far

- Maximum likelihood is a good way of estimating a grammar
- Maximum likelihood estimation of a PCFG from a treebank is easy *if the trees are accurate*
- But real language has many more dependencies than treebank grammar describes

$\Rightarrow$ relative frequency estimator not MLE

  – Make non-local dependencies local by splitting categories

  $\Rightarrow$ Astronomical number of possible categories

- Or find some way of dealing with non-local dependencies . . .

Exponential models

- Rules are not independent $\Rightarrow$ in general $Z \neq 1$
- Models with dependencies between features are called *exponential models*
  - Universe $\mathcal{T}$ (set of all possible parse trees)
  - Features $f = (f_1, \ldots, f_m)$ ($f_j(t)$ = value of j feature on $t \in \mathcal{T}$)
  - Feature weights $w = (w_1, \ldots, w_m)$

$$P(t) = \frac{1}{Z} \exp w \cdot f(t)$$

$$Z = \sum_{t' \in \mathcal{T}} \exp w \cdot f(t')$$

Hint: Think of $\exp w \cdot f(t)$ as unnormalized probability of t

PCFGs are exponential models

$\mathcal{T}$ = set of all trees generated by PCFG G

$f_j(t)$ = number of times the jth rule is used in $t \in \mathcal{T}$

$p(r_j)$ = probability of jth rule in G

Set weight $w_j = \log p(r_j)$

$$f\left(\begin{array}{ccc} & S & \\ NP & & VP \\ | & & | \\ rice & & grows \end{array}\right) = [\underbrace{1}_{S\to NP\ VP}, \underbrace{1}_{NP\to rice}, \underbrace{0}_{NP\to bananas}, \underbrace{1}_{VP\to grows}, \underbrace{0}_{VP\to grow}]$$

$$P_w(t) = \prod_{j=1}^{m} p(r_j)^{f_j(t)} = \prod_{j=1}^{m} (\exp w_j)^{f_j(t)} = \exp(w \cdot f(t))$$

So *a PCFG is just a special kind of exponential model* with $Z = 1$.

Advantages of exponential models

- Exponential models are very flexible . . .
- Features $\mathbf{f}$ can be *any function of parses* . . .
  - whether a particular structure occurs in a parse
  - conjunctions of prosodic and syntactic structure
- Parses $\mathbf{t}$ need not be trees, but *can be anything at all*
  - Feature structures (LFG, HPSG)
  - Minimalist derivations
- Exponential models are related to other popular models
  - Harmony theory and optimality theory
  - Maxent models

Modeling dependencies

- It's usually difficult to design a PCFG model that captures a particular set of dependencies
  - probability of the tree must be broken down into a product of *conditional probability distributions*
  - non-local dependencies must be expressed in terms of GPSG-style feature passing
- It's easy to make exponential models sensitive to new dependencies
  - add a new feature functions to existing feature functions
  - *figuring out what the right dependencies are is hard, but incorporating them into an exponential model is easy*

MLE of exponential models is hard

- An exponential model associates features $f(t) = (f_1(t), \ldots, f_m(t))$ with weights $w = (w_1, \ldots, w_m)$

$$P(t) = \frac{1}{Z} \exp w \cdot f(t)$$

$$Z = \sum_{t' \in \mathcal{T}} \exp w \cdot f(t')$$

- Given treebank $(t_1, \ldots, t_n)$, MLE chooses w to maximize $P(t_1) \times \ldots \times P(t_n)$, i.e., *make the treebank as likely as possible*
- Computing $P(t)$ requires the partition function Z
- Computing Z requires a sum over all parses $\mathcal{T}$ for *all sentences*

$\Rightarrow$ *computing MLE of an exponential parsing model seems very hard*

Conditional ML estimation

- Conditional ML estimation chooses feature weights to maximize $\mathsf{P}(\mathsf{t}_1|\mathsf{s}_1) \times \ldots \times \mathsf{P}(\mathsf{t}_n|\mathsf{s}_n)$, where s_i is string for t_i
  - choose feature weights to make t_i most likely relative to parses $\mathcal{T}(\mathsf{s}_i)$ for s_i

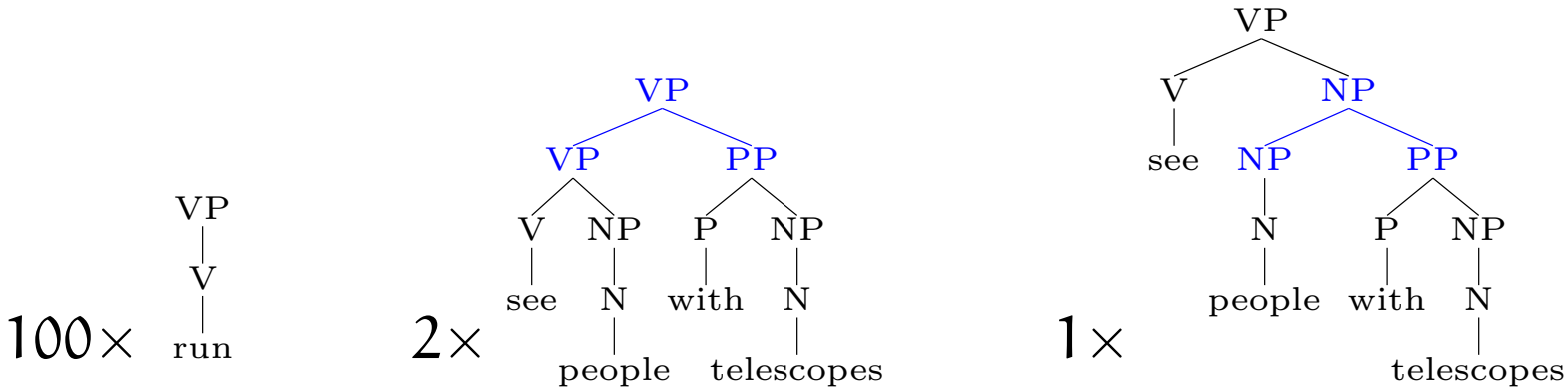
$\Rightarrow$ CMLE *doesn't involve other sentences*

$$\mathsf{P}(\mathsf{t}|\mathsf{s}) = \frac{1}{\mathsf{Z}(\mathsf{s})} \exp w \cdot \mathsf{f}(\mathsf{t})$$

$$\mathsf{Z}(\mathsf{s}) = \sum_{\mathsf{t}' \in \mathcal{T}(\mathsf{s})} \exp w \cdot \mathsf{f}(\mathsf{t}')$$

- CMLE "only" involves repeatedly parsing training data
- With "wrong" models, CMLE often produces a more accurate parser than joint MLE

Conditional vs joint MLE



$\ldots \times 2/105 \times \ldots$

$\ldots \times 2/7 \times \ldots$

$\ldots \times 1/7 \times \ldots$

$\ldots \times 1/7 \times \ldots$

Rule	**count**	**rel freq**	**rel freq**
VP → V	100	100/105	4/7
VP → V NP	3	3/105	1/7
VP → VP PP	2	2/105	**2/7**
NP → N	6	6/7	6/7
NP → NP PP	1	1/7	**1/7**

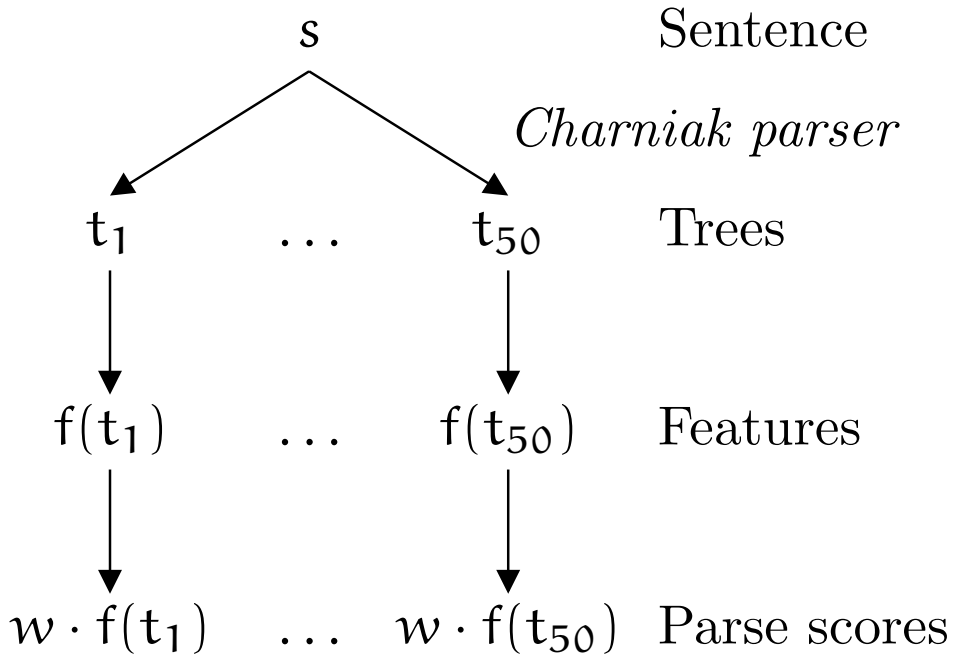
Conditional ML estimation

s	$f(\bar{t})$	$\{f(t) : t \in \mathcal{T}(s), t \neq \bar{t}(s)\}$
sentence 1	$(1,3,2)$	$(2,2,3)$ $(3,1,5)$ $(2,6,3)$
sentence 2	$(7,2,1)$	$(2,5,5)$
sentence 3	$(2,4,2)$	$(1,1,7)$ $(7,2,1)$
...	...	...

- Parser designer specifies *feature functions* $f = (f_1, \ldots, f_m)$
- A parser produces trees $\mathcal{T}(s)$ for each sentence s
- Treebank tells us correct tree $\bar{t}(s) \in \mathcal{T}(s)$ for sentence s
- Feature functions f apply to each tree $t \in \mathcal{T}(s)$, producing feature values $f(t) = (f_1(t), \ldots, f_m(t))$
- MCLE estimates feature weights $w = (w_1, \ldots, w_m)$

Coarse-to-fine parsing

- $Z(s)$ is still hard to compute $\Rightarrow$ make $\mathcal{T}(s)$ even smaller!
- Restrict attention to 50-best parses produced by Charniak parser (a good PCFG-based parser)
- Exponential model is trained using CMLE to pick out best parse from Charniak's 50-best parses



Parser evaluation

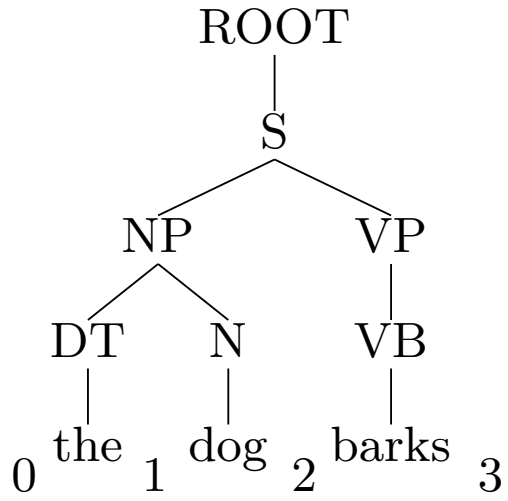
- A node's *edge* is its label and beginning and ending *string positions*
- $\mathsf{E}(t)$ is the set of edges associated with a tree t
- If t is a parse tree and $\bar{t}$ is the correct tree, then

  precision $\quad \mathsf{P}_{\bar{t}}(t) = |\mathsf{E}(t)|/|\mathsf{E}(t) \cap \mathsf{E}(\bar{t})|$

  recall $\quad \mathsf{R}_{\bar{t}}(t) = |\mathsf{E}(\bar{t})|/|\mathsf{E}(t) \cap \mathsf{E}(\bar{t})|$

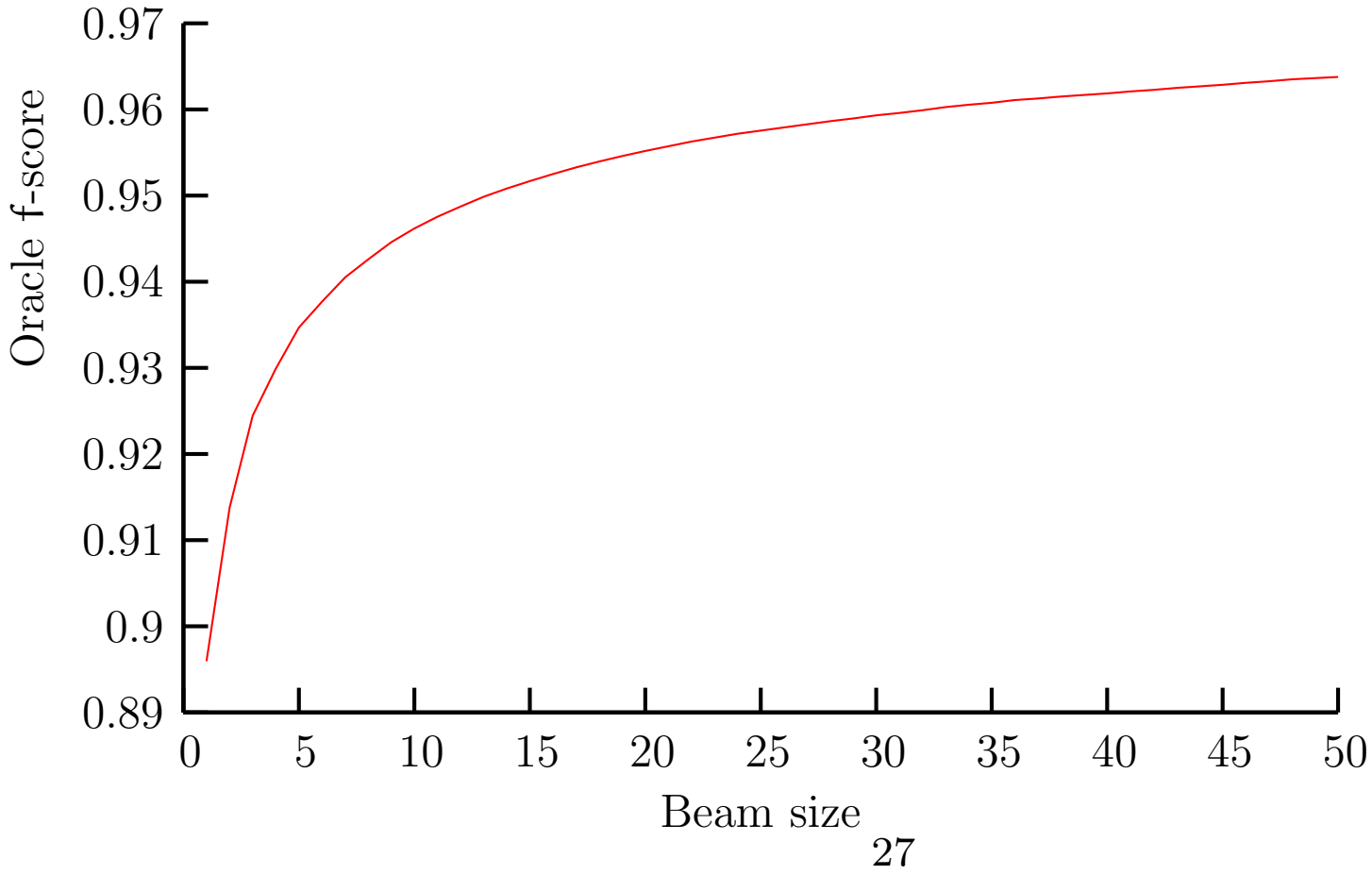
  f-*score* $\quad \mathsf{F}_{\bar{t}}(t) = 2/(\mathsf{P}_{\bar{t}}(t)^{-1} + \mathsf{R}_{\bar{y}}(t)^{-1})$ (geometric mean of P and R)

Edges		
(0	NP	2)
(2	VP	3)
(0	S	3)



Performance of Charniak parser

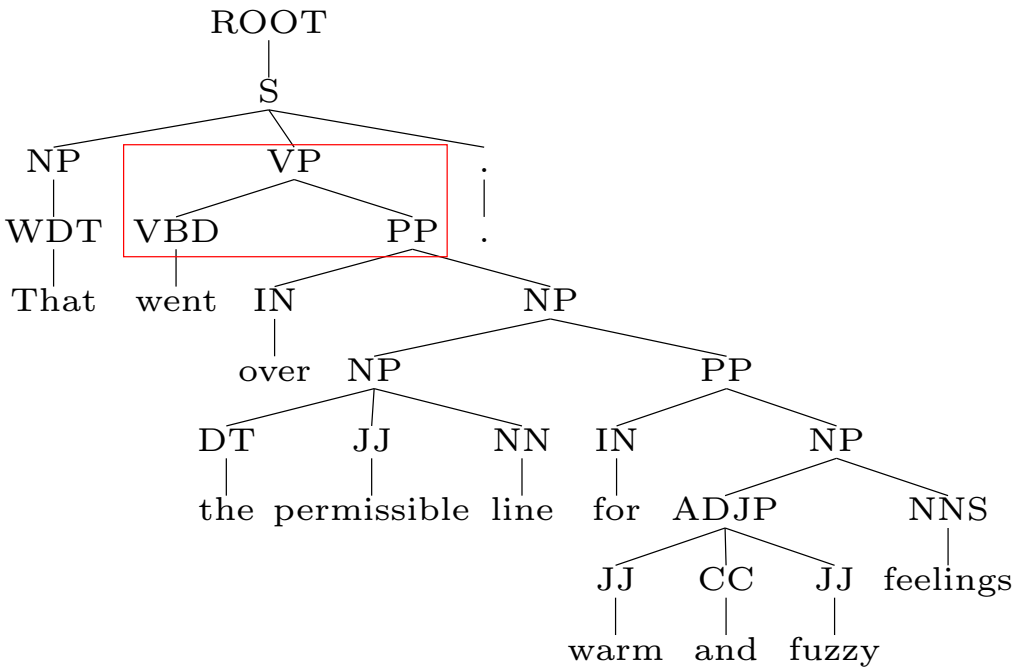
- F-score of Charniak's most probable parse = 0.896 (cross-validated on PTB sections 2-19)
- Oracle f-score of Charniak's 50-best parses = 0.965 (66% redn)



Expt 1: Only "old" features

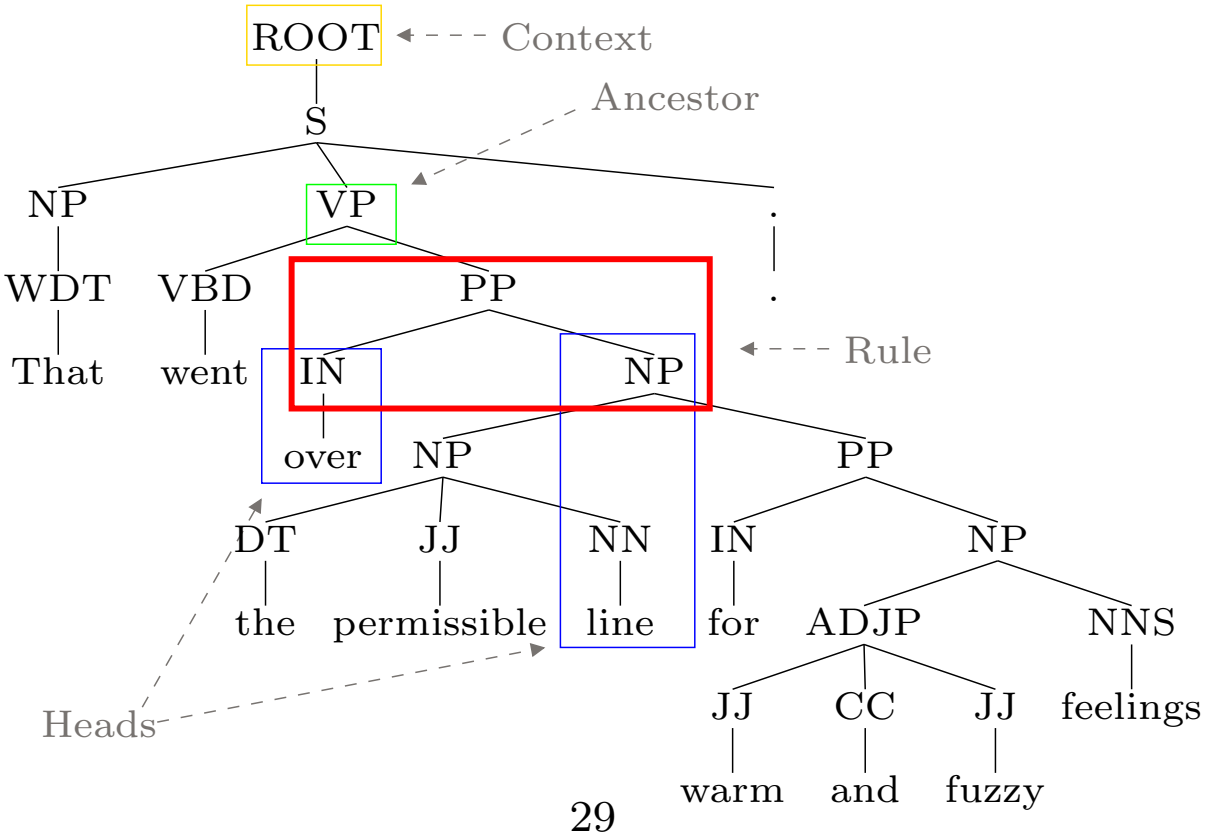
- Features: 1 *log Charniak probability*, 10,124 Rule features
- Charniak's parser already conditions on local trees!
- Feature selection: features must vary on 5 or more sentences
- Results: f-score = 0.894; baseline = 0.890; $\approx 4\%$ error reduction

$\Rightarrow$ *discriminative training alone can improve accuracy*



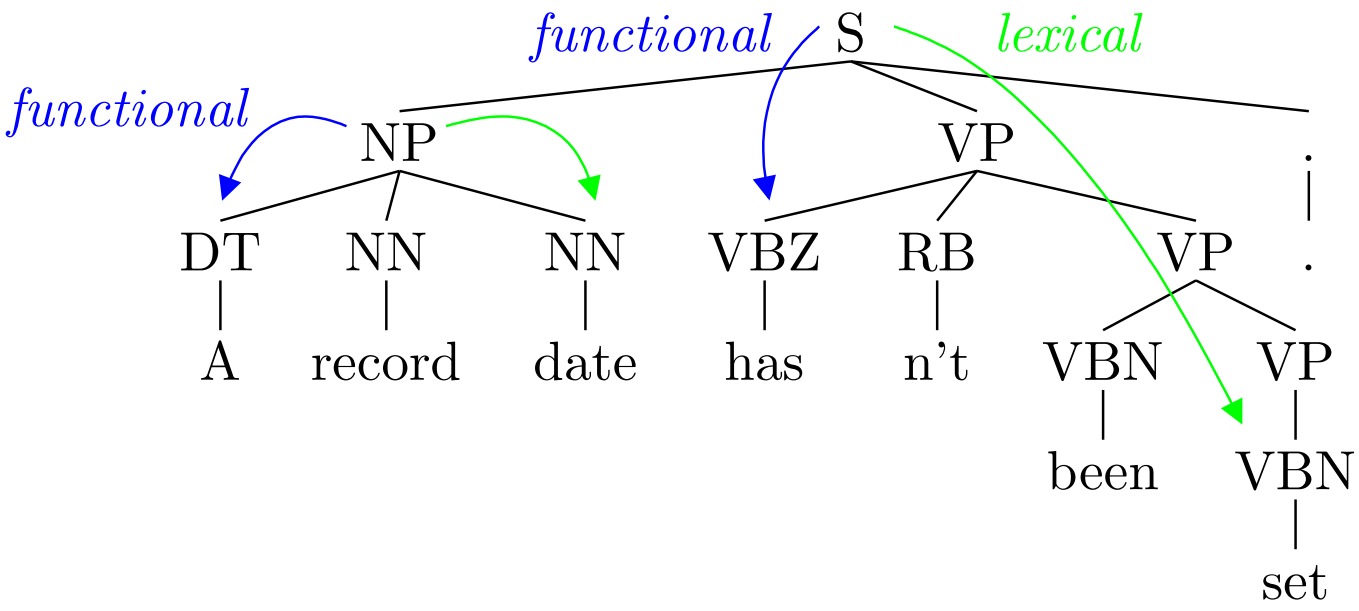
Lexicalized and parent-annotated rules

- *Lexicalization* associates each constituent with its head
- *Ancestor annotation* provides a little "vertical context"
- *Context annotation* indicates constructions that only occur in main clause (c.f., Emonds)



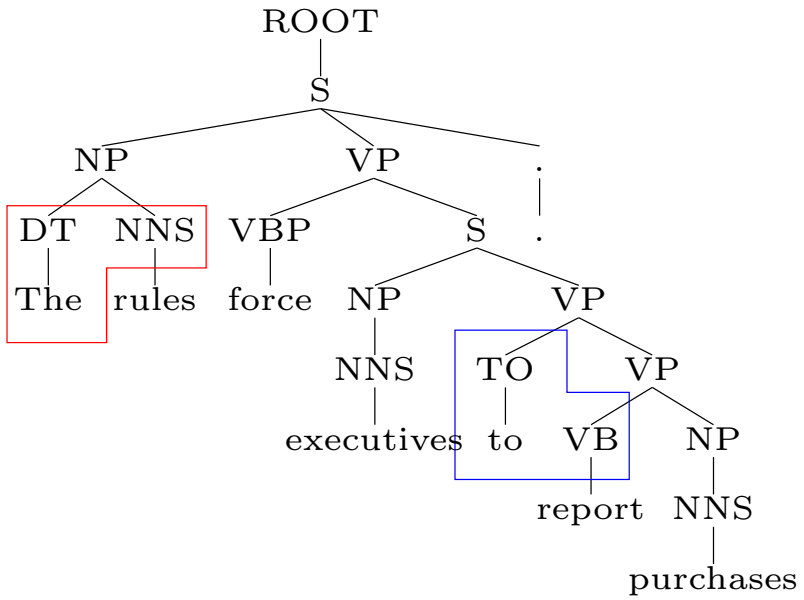
Functional and lexical heads

- There are at least two sensible notions of head (c.f., Grimshaw)
  - *Functional heads:* determiners of NPs, auxilary verbs of VPs, etc.
  - *Lexical heads:* rightmost Ns of NPs, main verbs in VPs, etc.
- In a Maxent model, it is easy to use both!



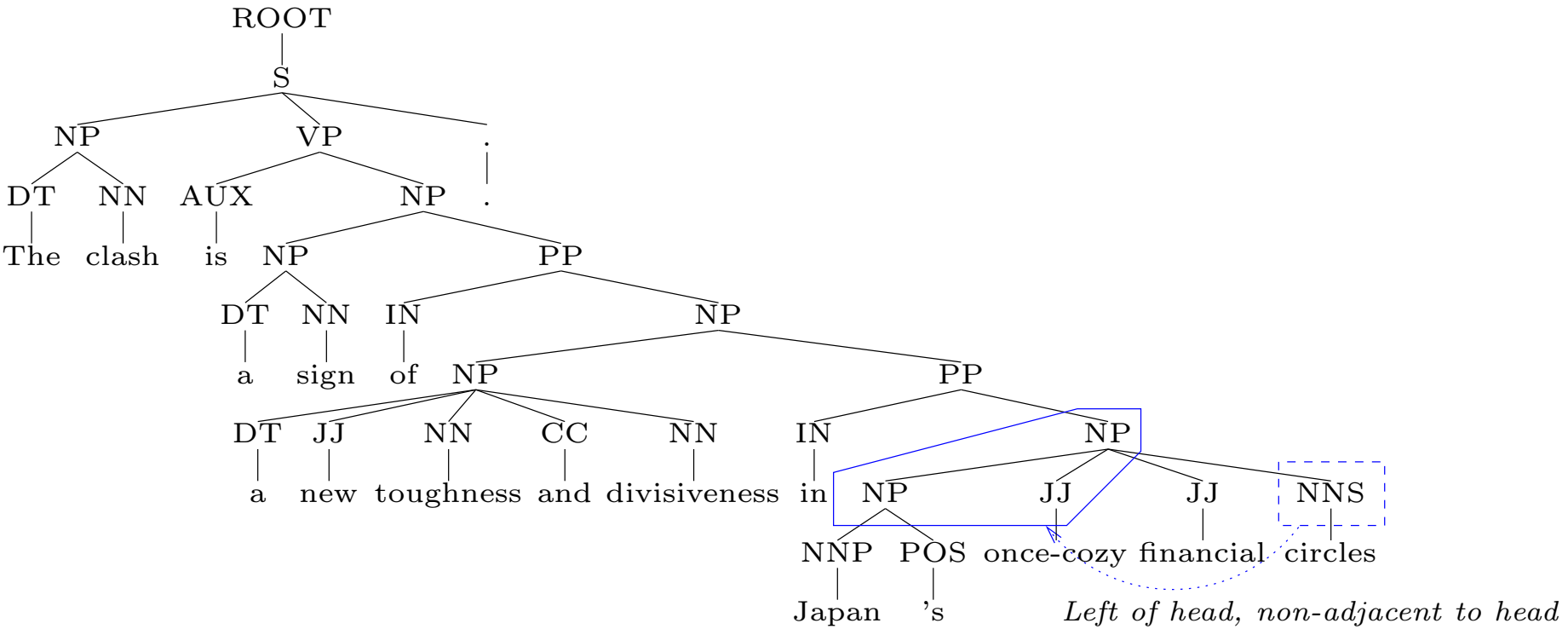
Functional-lexical head dependencies

- The SynSemHeads features collect pairs of functional and lexical heads of phrases
- This captures number agreement in NPs and aspects of other head-to-head dependencies



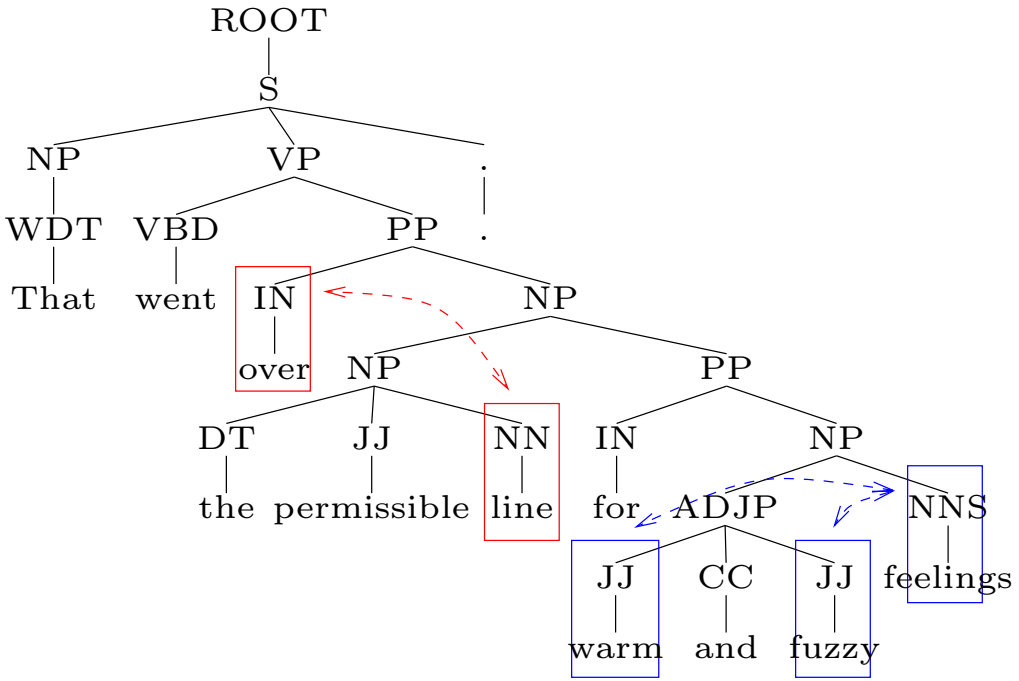
n-gram rule features generalize rules

- Collects *adjacent constituents* in a local tree
- Also includes *relationship to head*
- Constituents can be *ancestor-annotated* and *lexicalized*



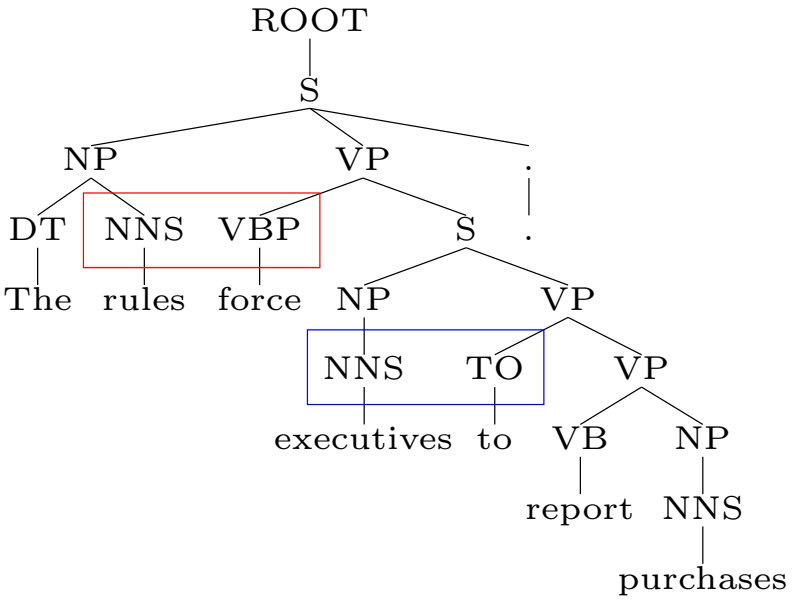
Head to head dependencies

- Head-to-head dependencies track the function-argument dependencies in a tree
- Co-ordination leads to phrases with multiple heads or functors
- Parameterized by head type and number of governors to include



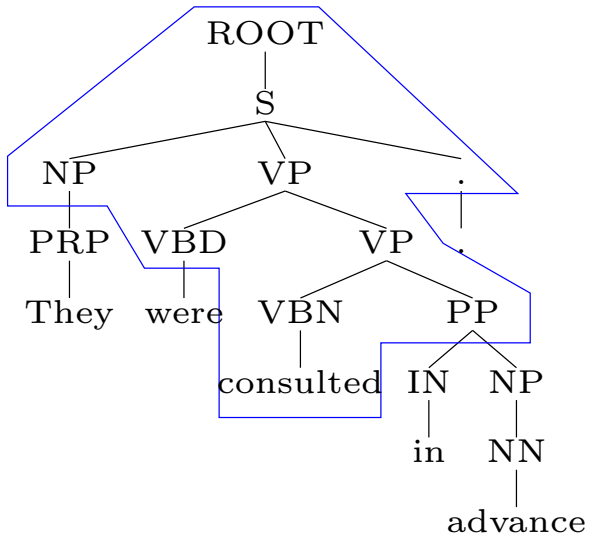
Subject-Verb Agreement

- The SubjVerbAgr features are the POS of the subject NP's lexical head and the VP's functional head



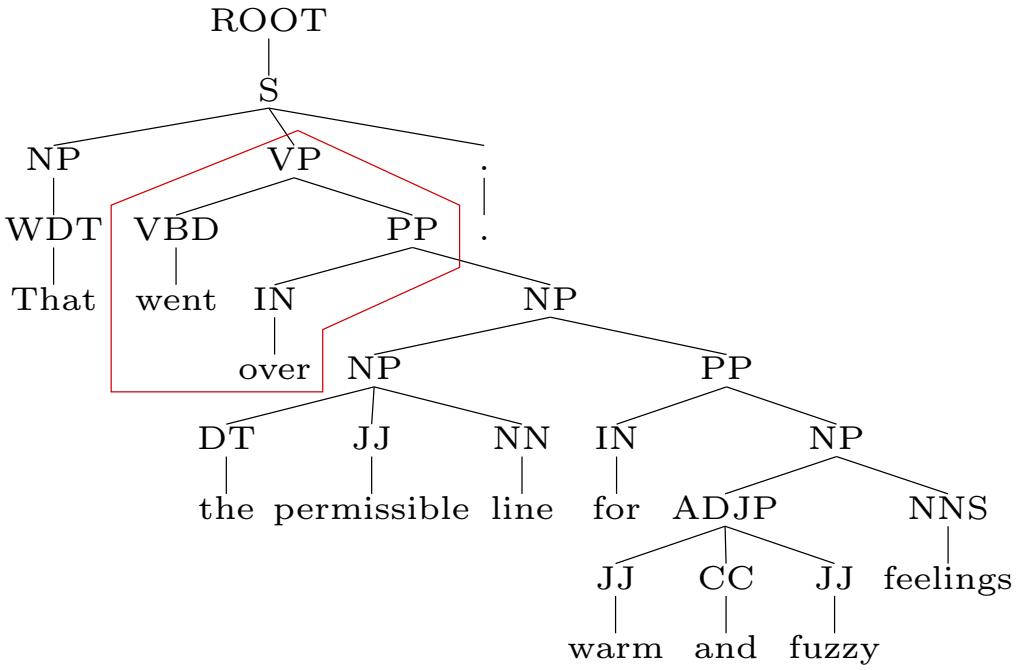
Head trees record all dependencies

- Head trees consist of a (lexical) head, all of its projections and (optionally) all of the siblings of these nodes
- These correspond roughly to TAG elementary trees



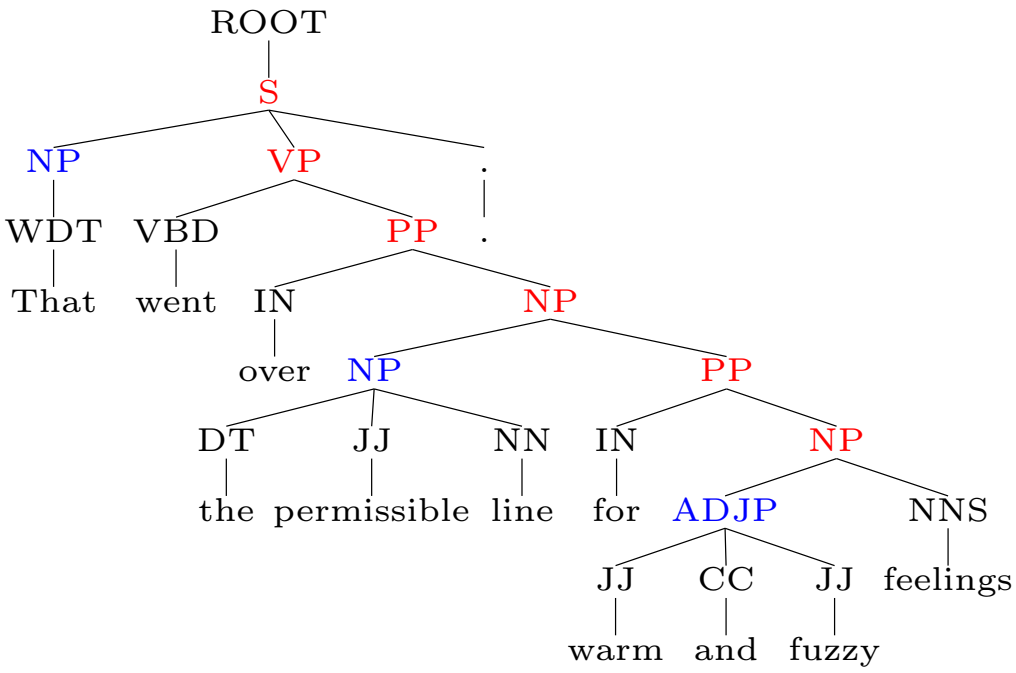
Tree n-gram

- A tree n-gram are tree fragments that connect sequences of adjacent n words



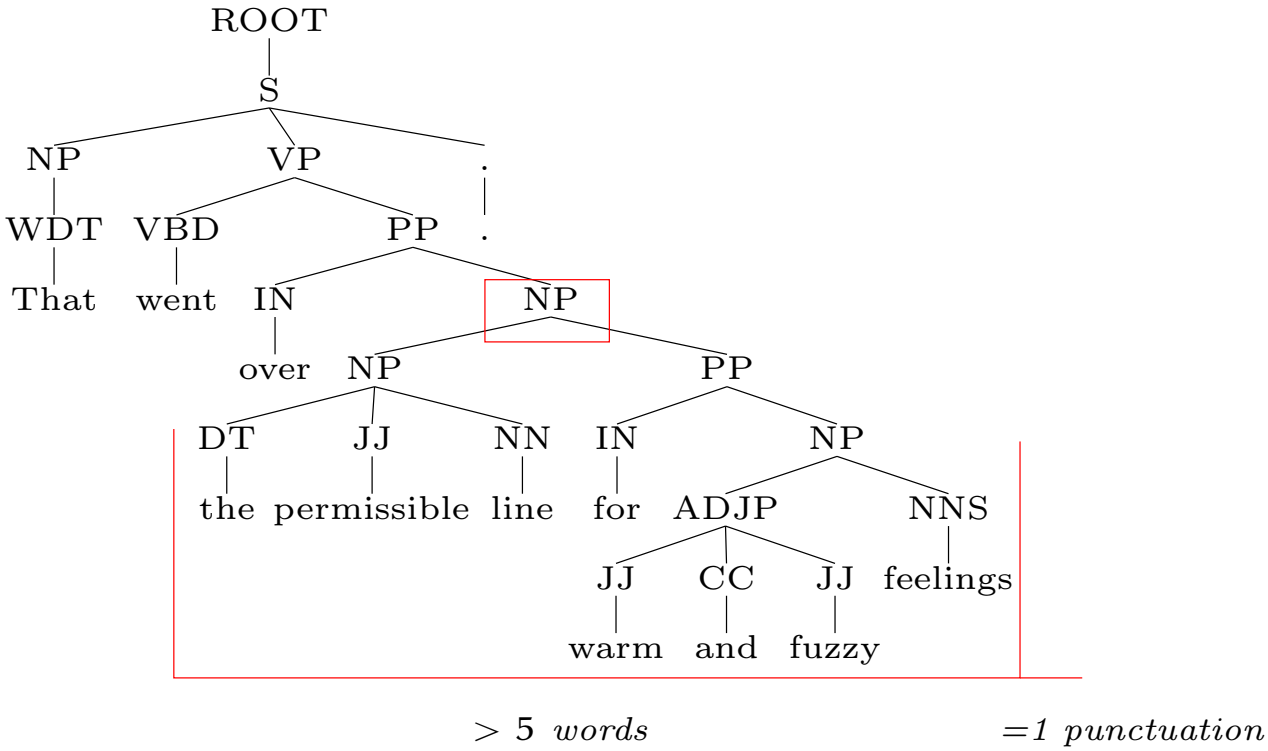
Rightmost branch bias

- The RightBranch feature's value is the number of nodes on the right-most branch (ignoring punctuation)
- Reflects the tendancy toward right branching in English
- Only 2 different features, but very useful in final model!



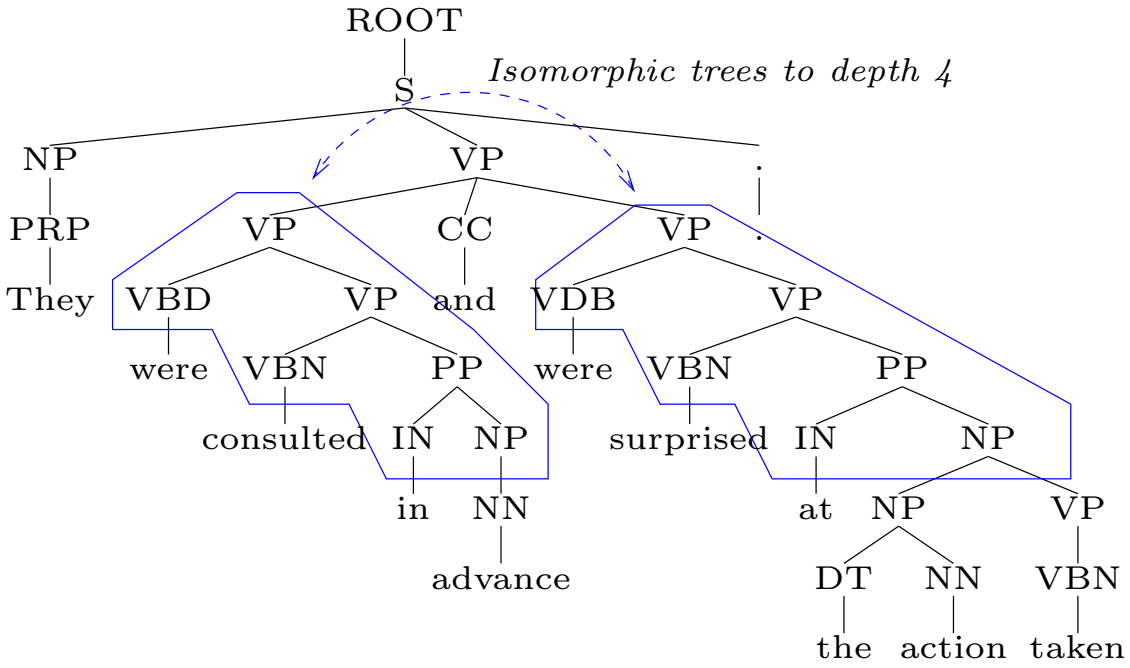
Constituent Heavyness and location

- Heavyness measures the constituent's category, its (binned) size and (binned) closeness to the end of the sentence



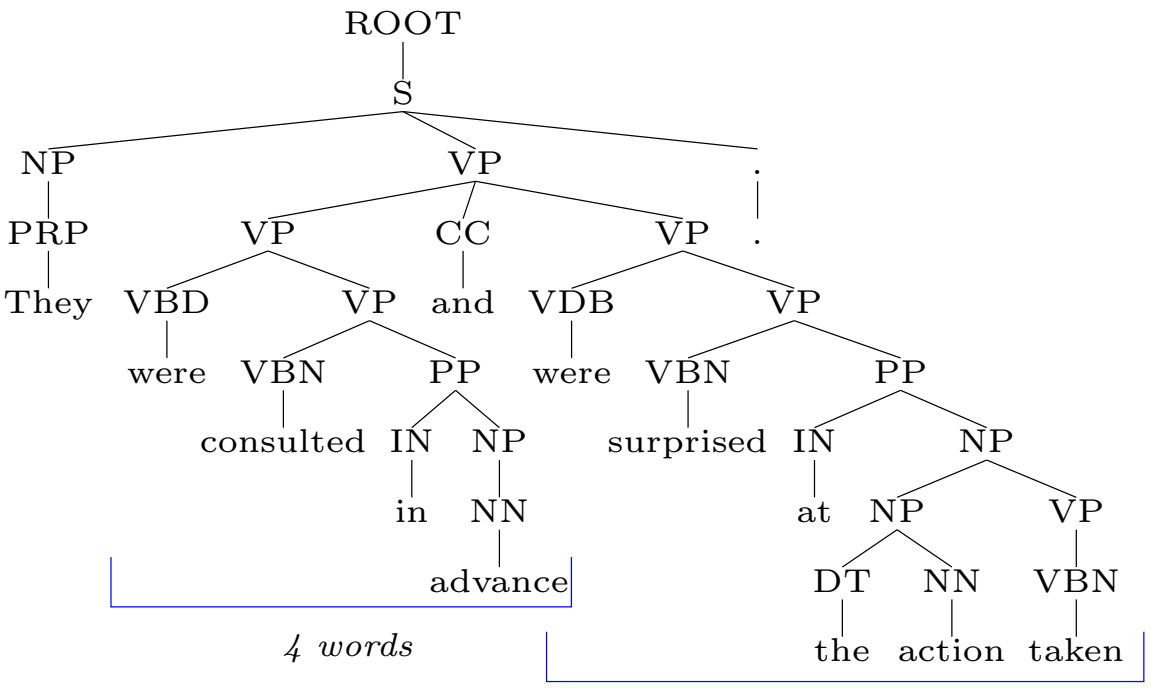
Coordination parallelism (1)

- The CoPar feature indicates the depth to which adjacent conjuncts are parallel



Coordination parallelism (2)

- The CoLenPar feature indicates the difference in length in adjacent conjuncts and whether this pair contains the last conjunct.



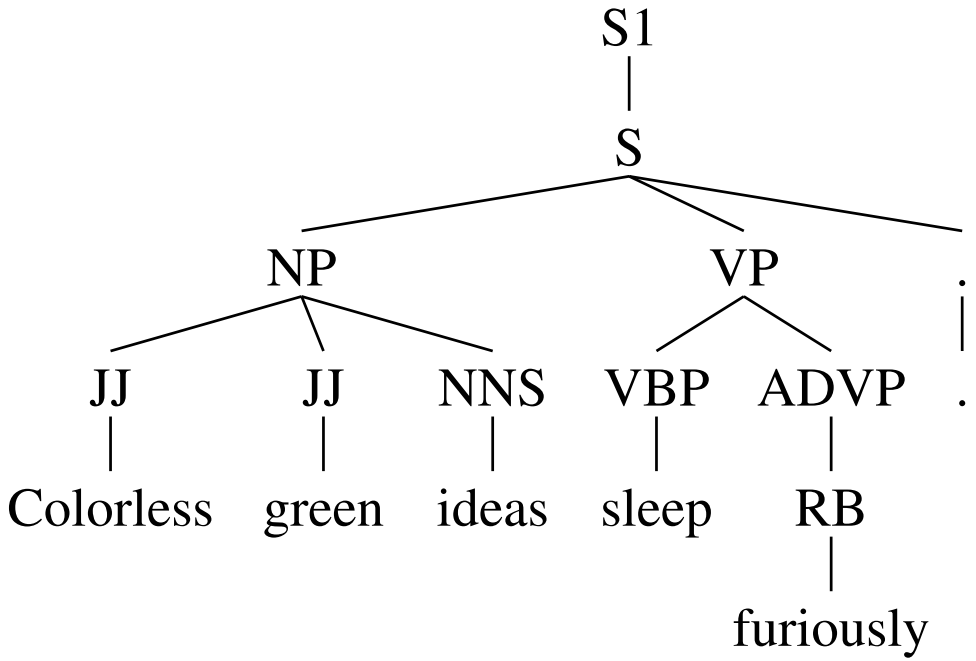
CoLenPar feature: (2,true) 6 words

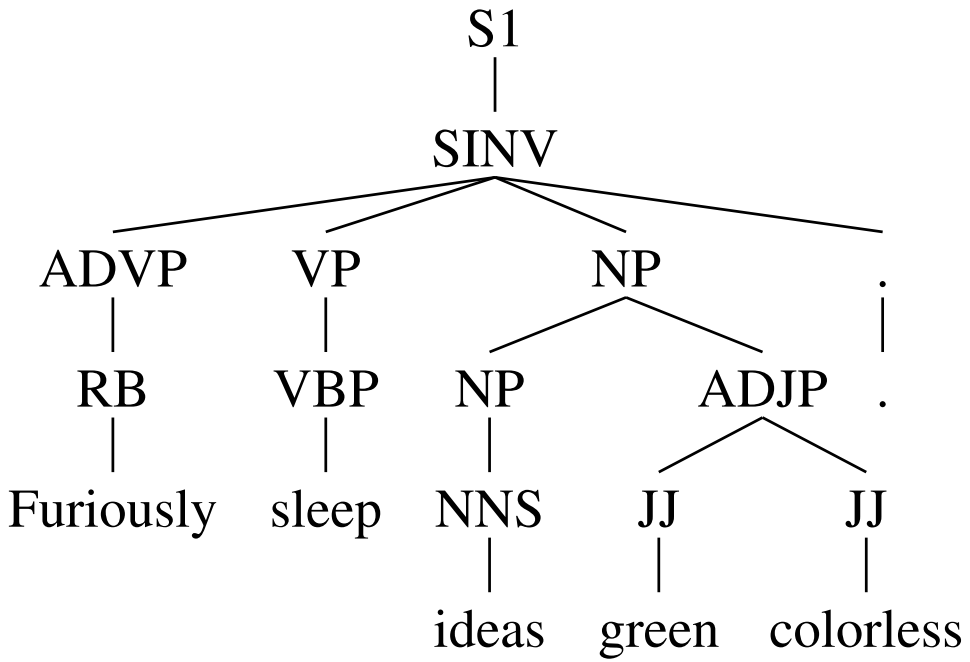
Experimental results with all features

- Features must vary on parses of at least 5 sentences in training data
- In this experiment, 724,550 features
- Gaussian regularization, adjusted via cross-validation on section 23
- *f-score on section 23* = 0.912 (15% error reduction over Charniak parser)

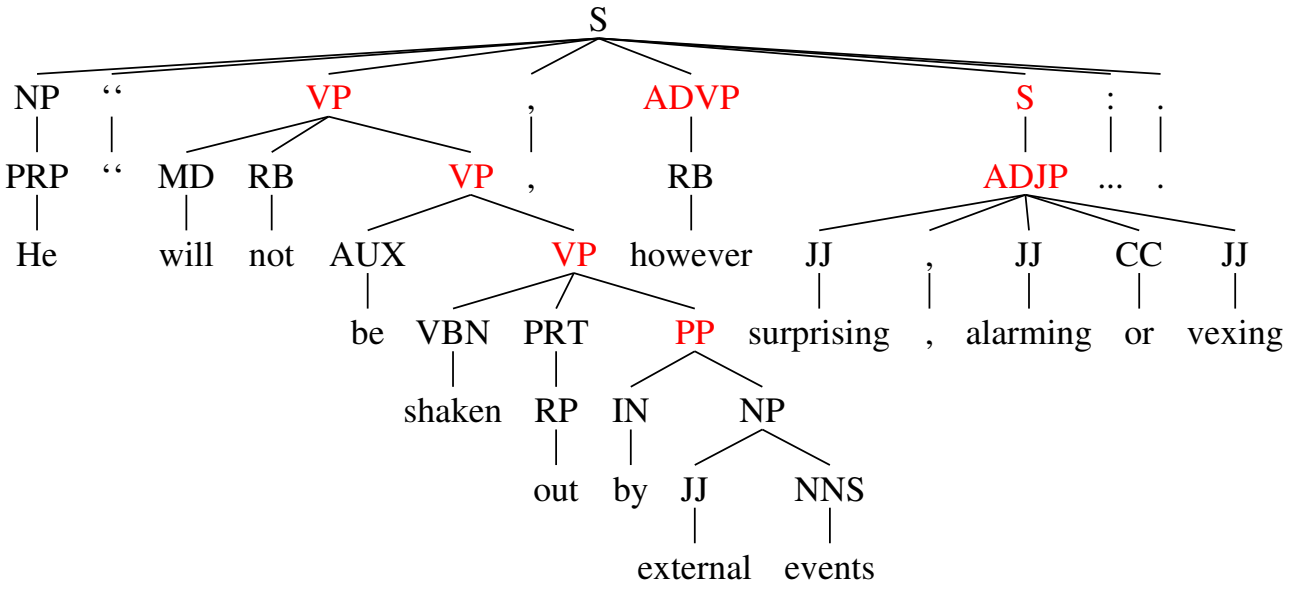
Conclusion

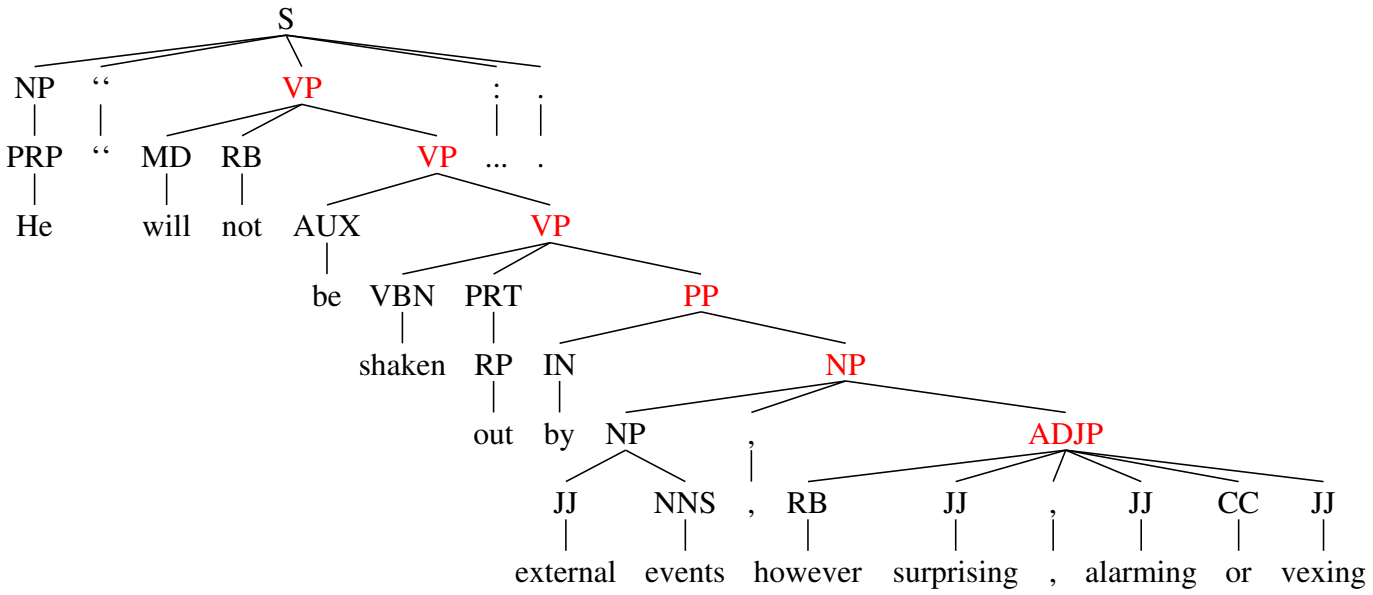
- It's possible to build (moderately) accurate, broad-coverage parsers
- Generative parsing models are easy to estimate, but make questionable independence assumptions
- Exponential models don't assume independence, so it's easy to add new features, but are difficult to estimate
- Coarse-to-fine conditional MLE for exponential models is a compromise
  - flexibility of exponential models
  - possible to estimate from treebank data
- Gives the currently best-reported parsing accuracy results

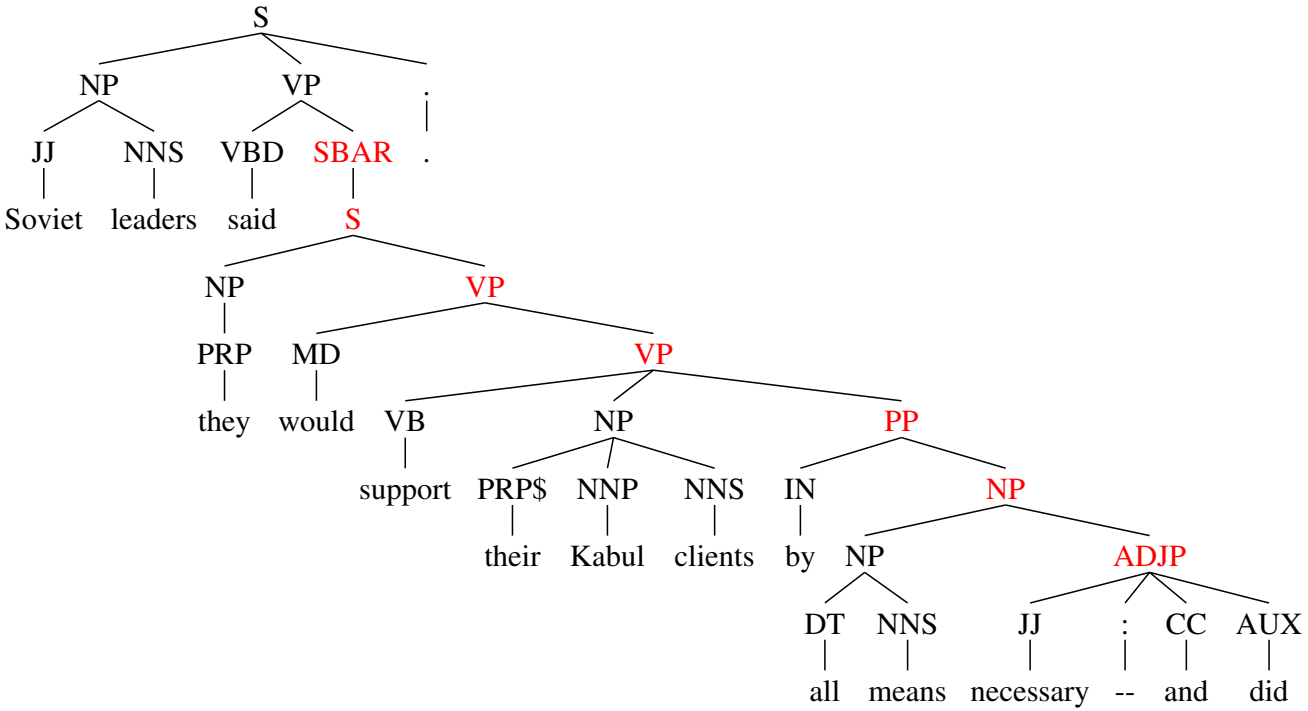

S1
S
NP
VP
.
JJ
JJ
NNS
VBP
ADVP
.
Colorless
green
ideas
sleep
RB
furiously

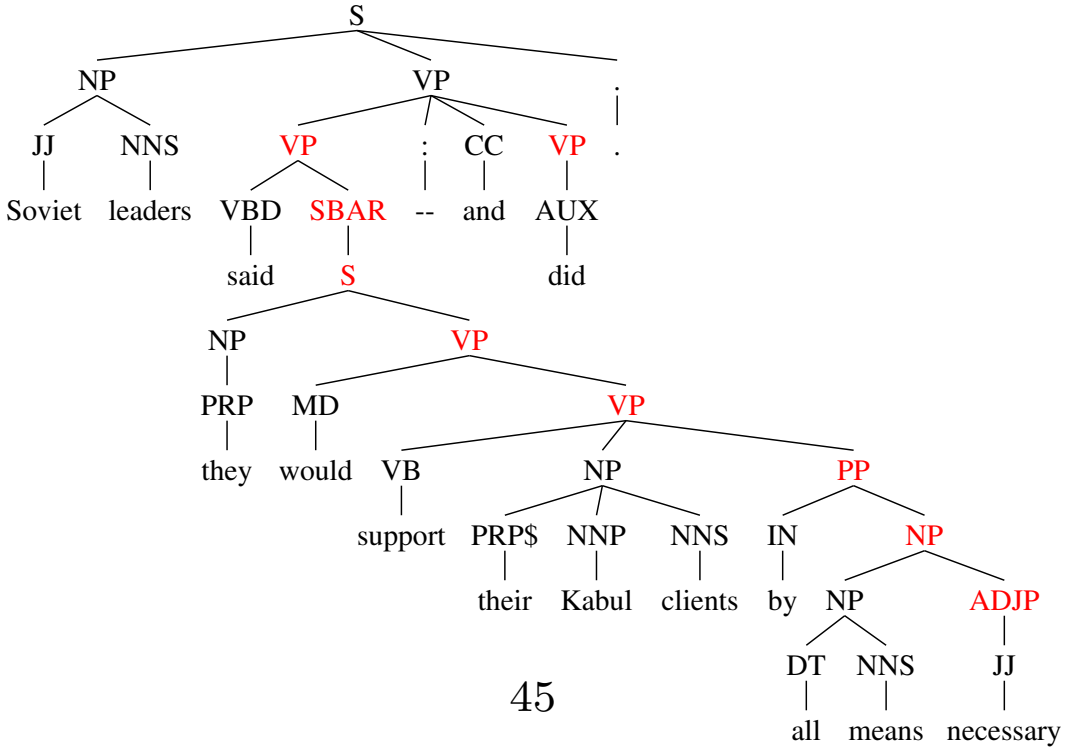

S1
SINV
ADVP
VP
NP
.
RB
VBP
NP
ADJP
.
Furiously
sleep
NNS
JJ
JJ
ideas
green
colorless

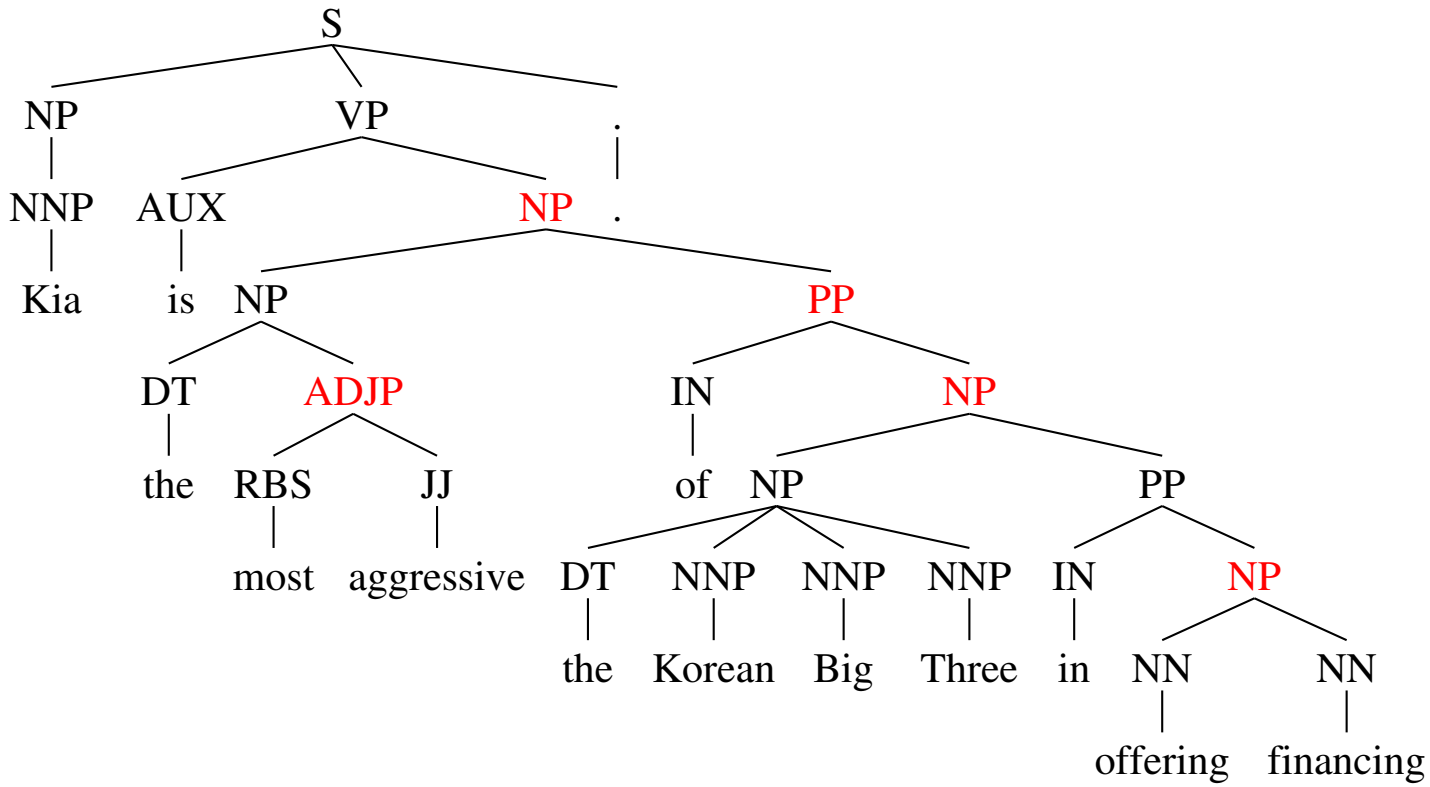
Sample parser errors

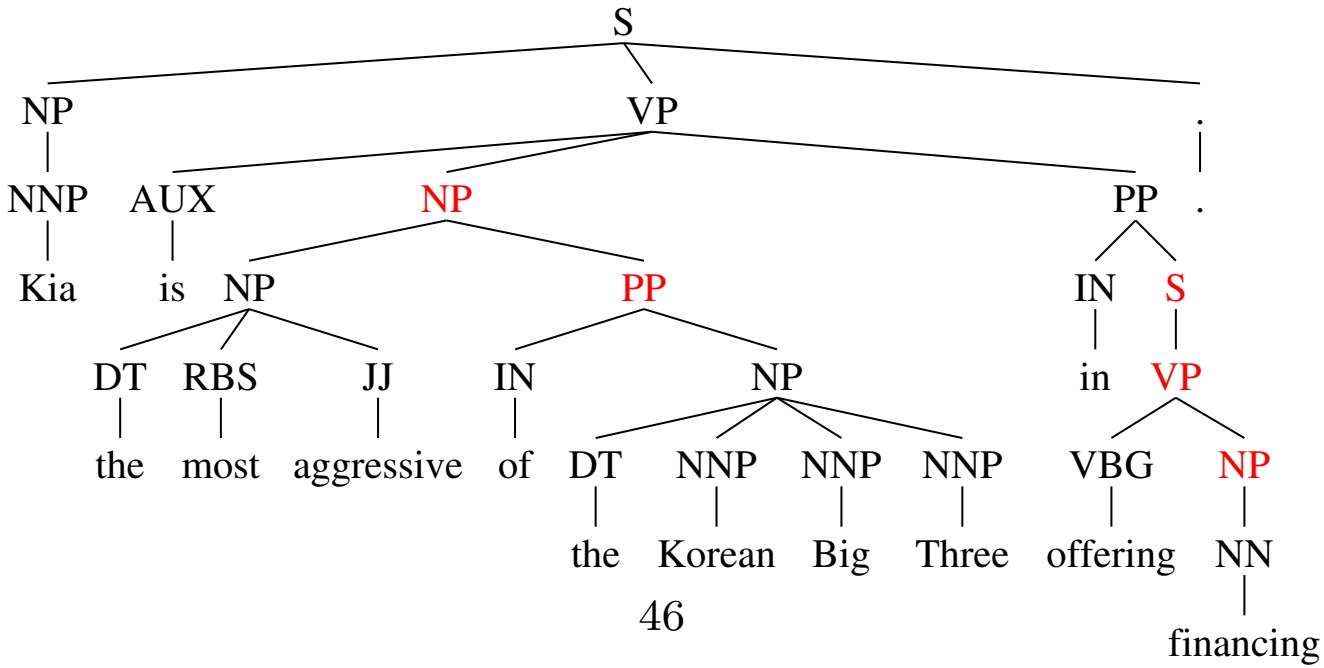


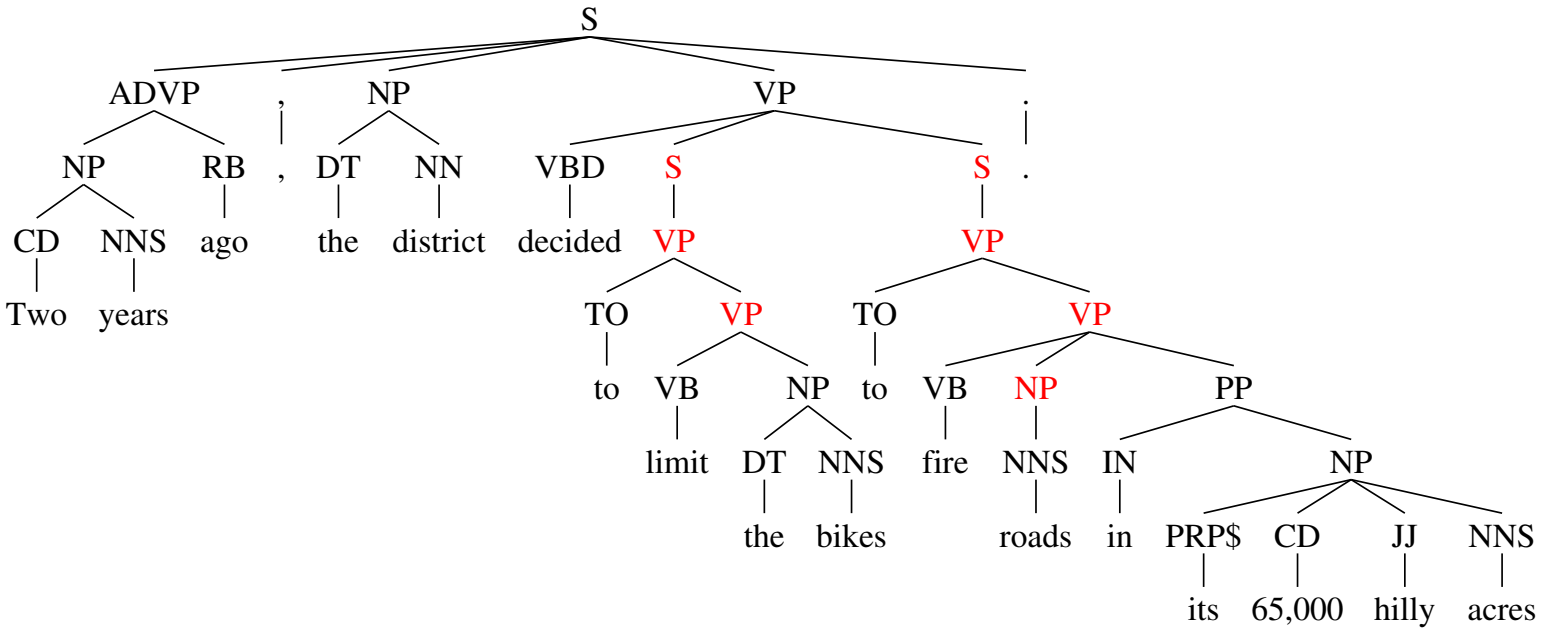


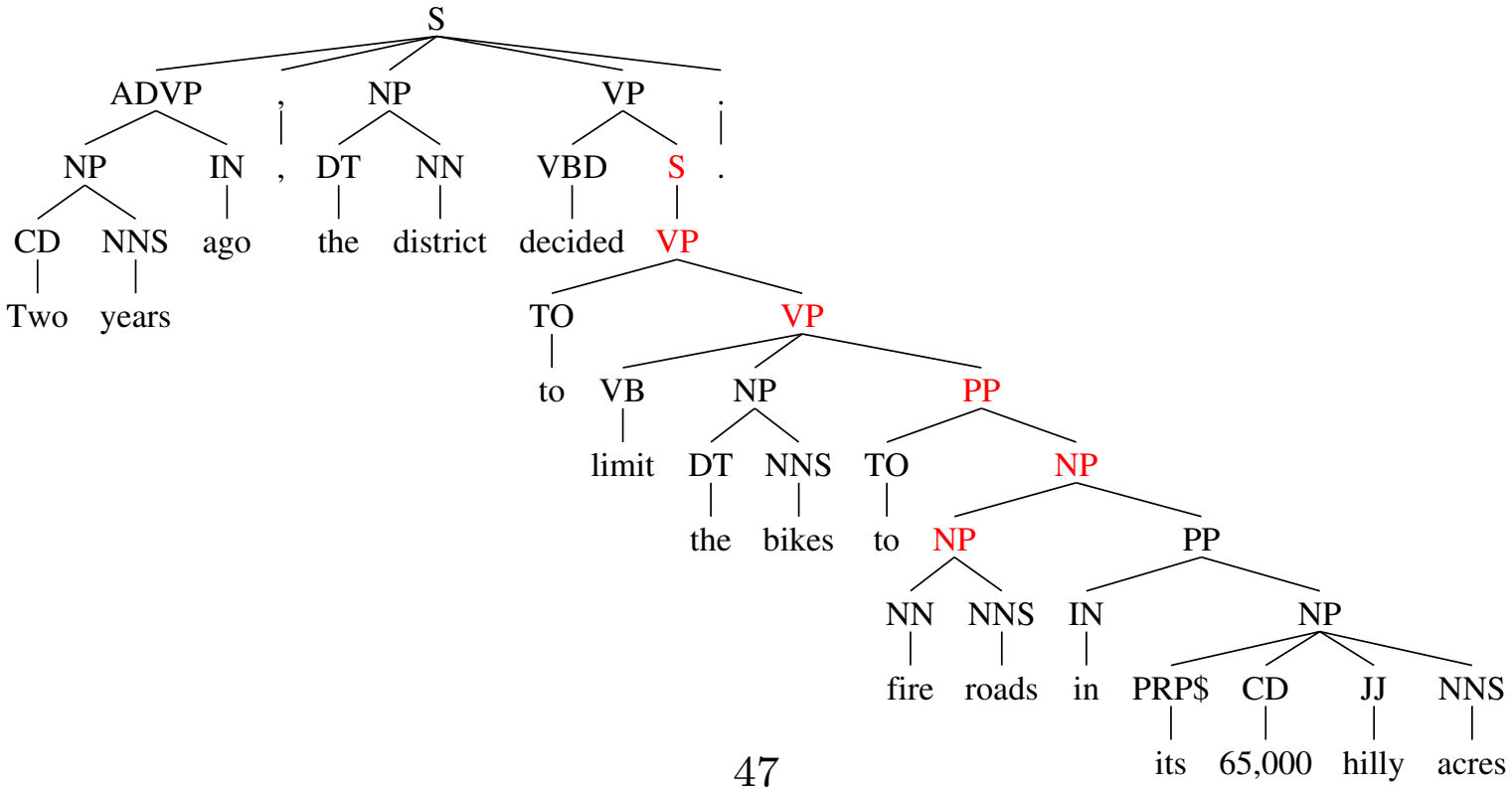

S
NP
VP
.
JJ
NNS
VBD
SBAR
.
Soviet
leaders
said
S
NP
VP
PRP
MD
VP
they
would
VB
NP
PP
support
PRP$
NNP
NNS
IN
NP
their
Kabul
clients
by
NP
ADJP
DT
NNS
JJ
:
CC
AUX
all
means
necessary
--
and
did

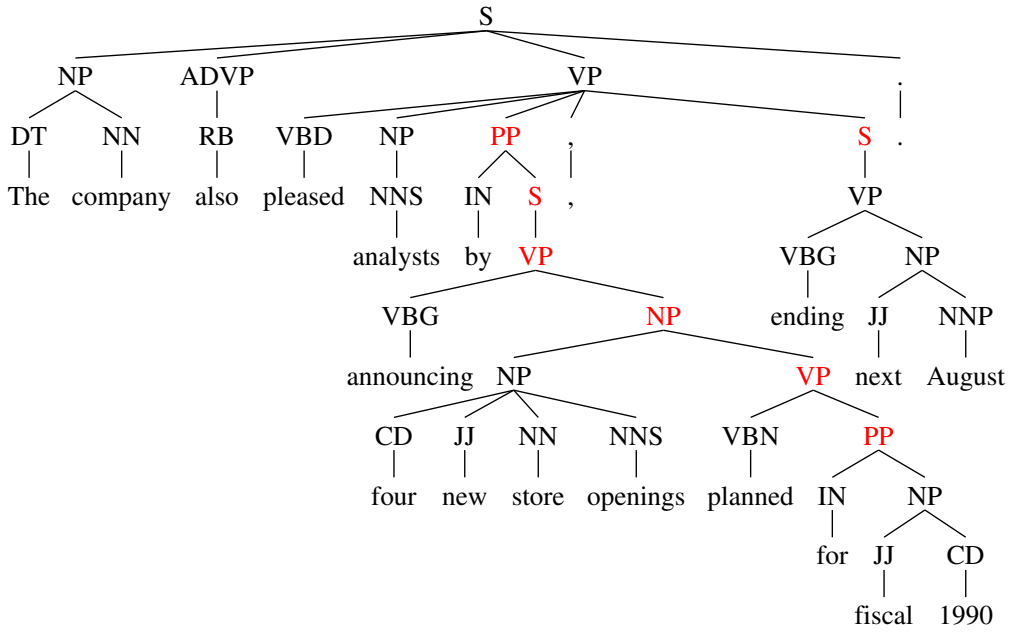

S
NP
VP
.
JJ
NNS
VP
:
CC
VP
.
Soviet
leaders
VBD
SBAR
--
and
AUX
said
S
did
NP
VP
PRP
MD
VP
they
would
VB
NP
PP
support
PRP$
NNP
NNS
IN
NP
their
Kabul
clients
by
NP
ADJP
DT
NNS
JJ
all
means
necessary

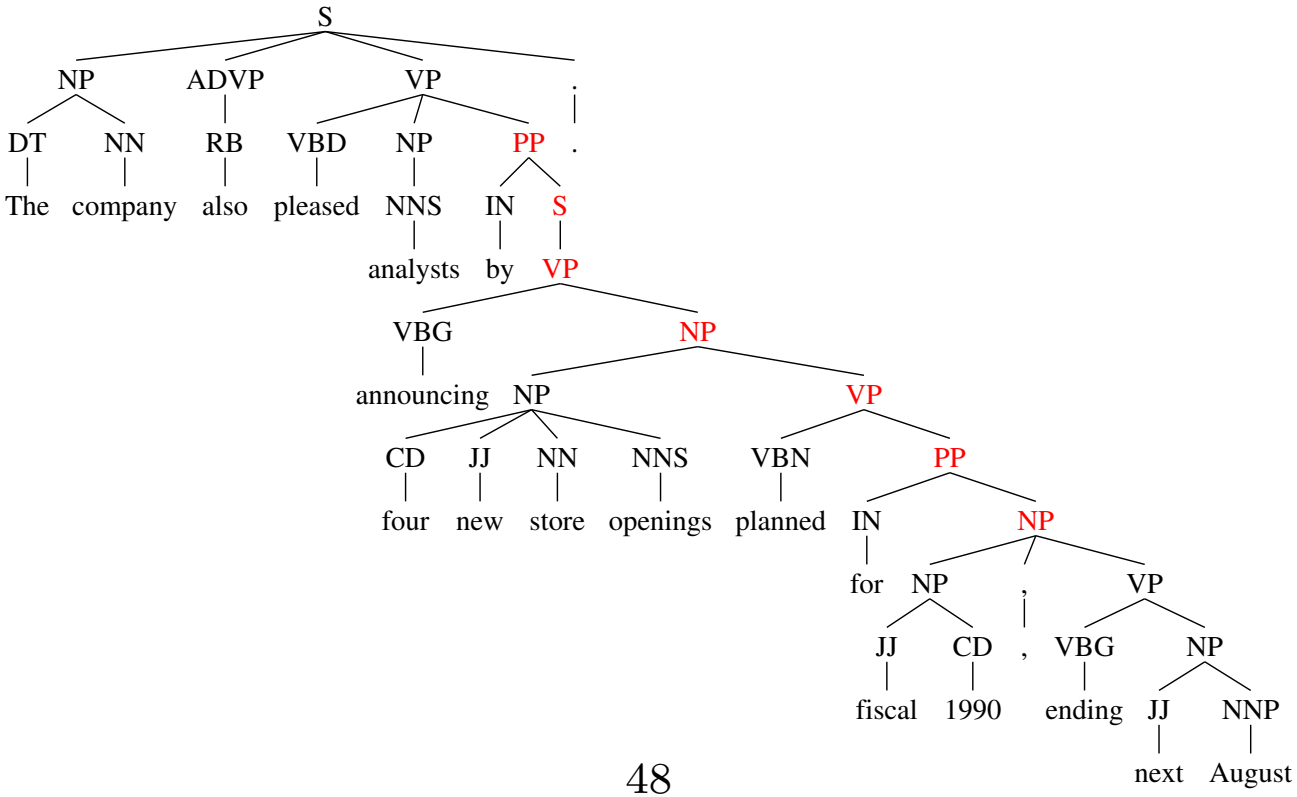

S
NP
VP
.
NNP
AUX
NP
.
Kia
is
NP
PP
DT
ADJP
IN
NP
the
RBS
JJ
of
NP
PP
most
aggressive
DT
NNP
NNP
NNP
IN
NP
the
Korean
Big
Three
in
NN
NN
offering
financing

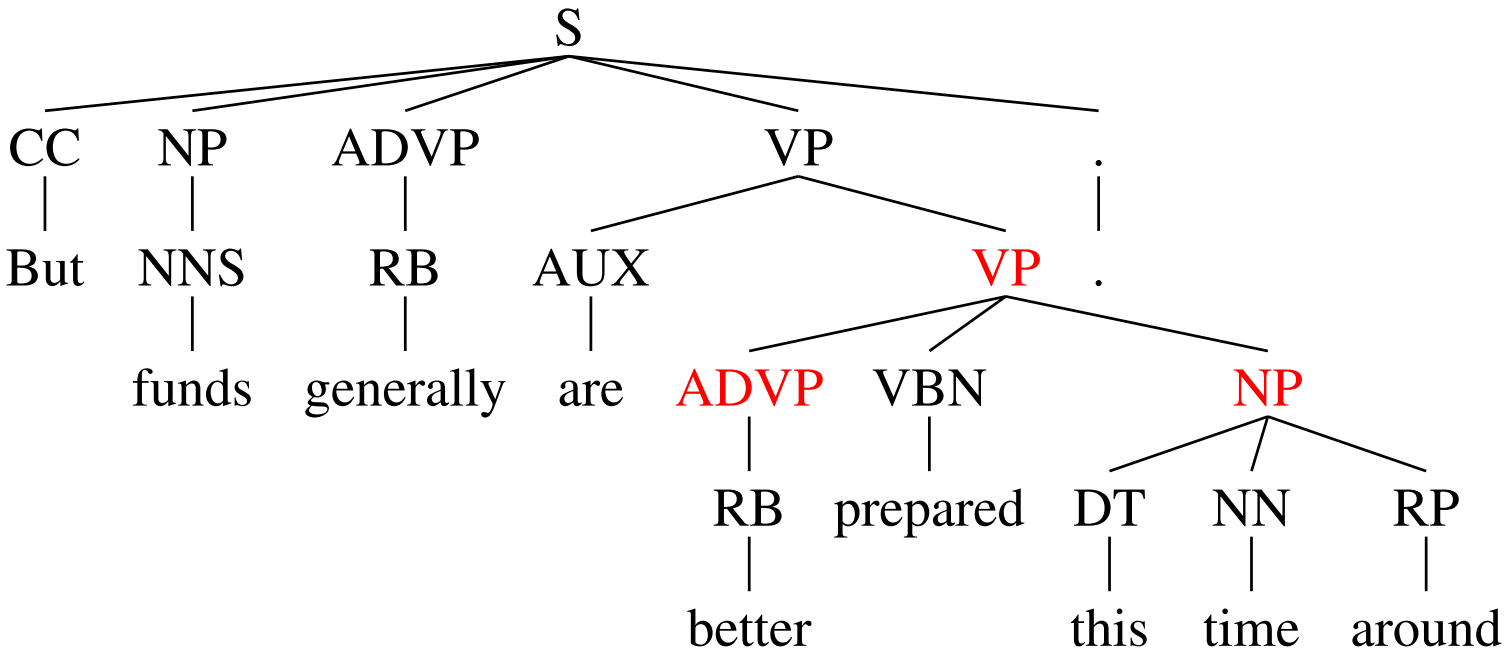

S
NP
VP
.
NNP
AUX
NP
PP
.
Kia
is
NP
PP
IN
S
DT
RBS
JJ
IN
NP
in
VP
the
most
aggressive
of
DT
NNP
NNP
NNP
VBG
NP
the
Korean
Big
Three
offering
NN
financing

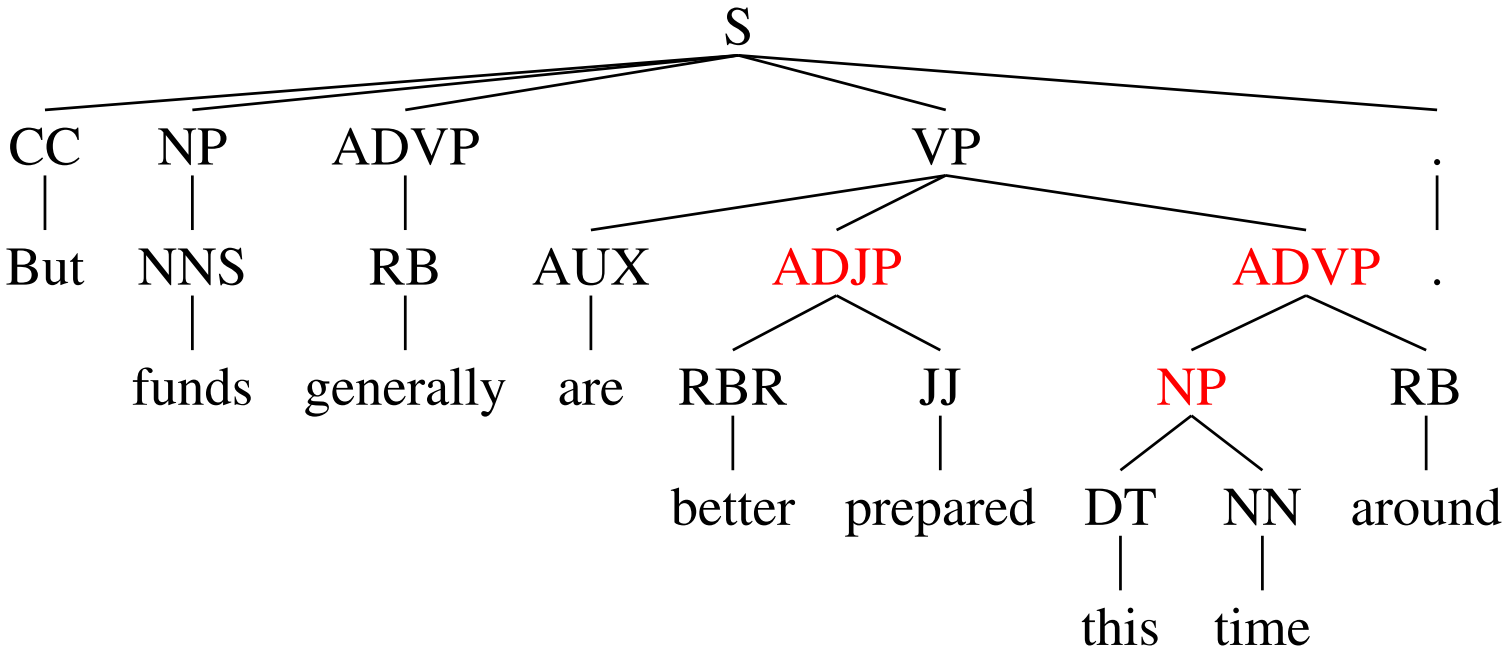

S
ADVP , NP VP .
NP RB , DT NN VBD S S .
CD NNS ago the district decided VP VP
Two years TO VP TO VP
to VB NP to VB NP PP
limit DT NNS fire NNS IN NP
the bikes roads in PRP$ CD JJ NNS
its 65,000 hilly acres

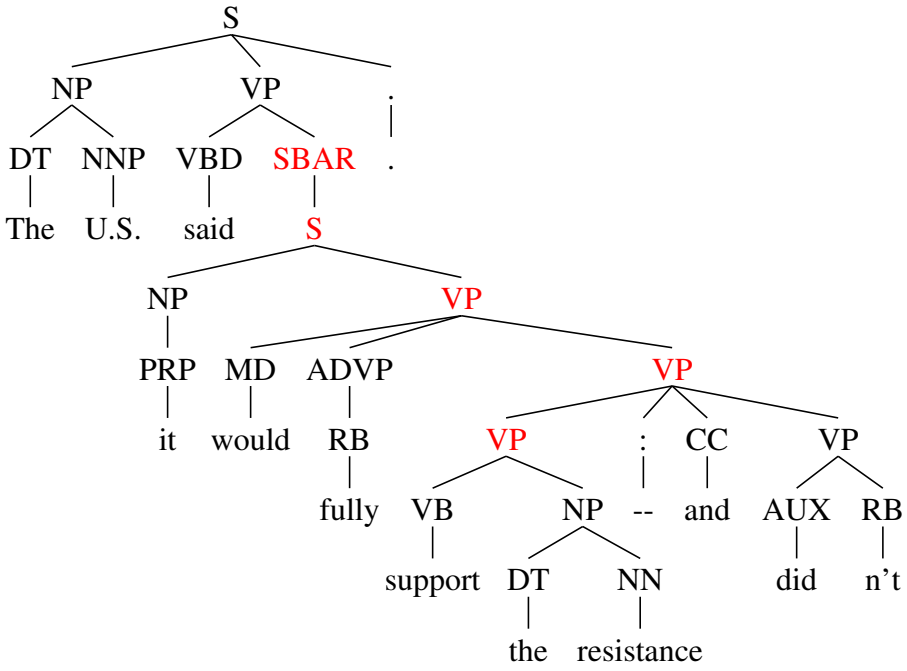

S
ADVP , NP VP .
NP IN , DT NN VBD S .
CD NNS ago the district decided VP
Two years TO VP
to VB NP PP
limit DT NNS TO NP
the bikes to NP PP
NN NNS IN NP
fire roads in PRP$ CD JJ NNS
its 65,000 hilly acres

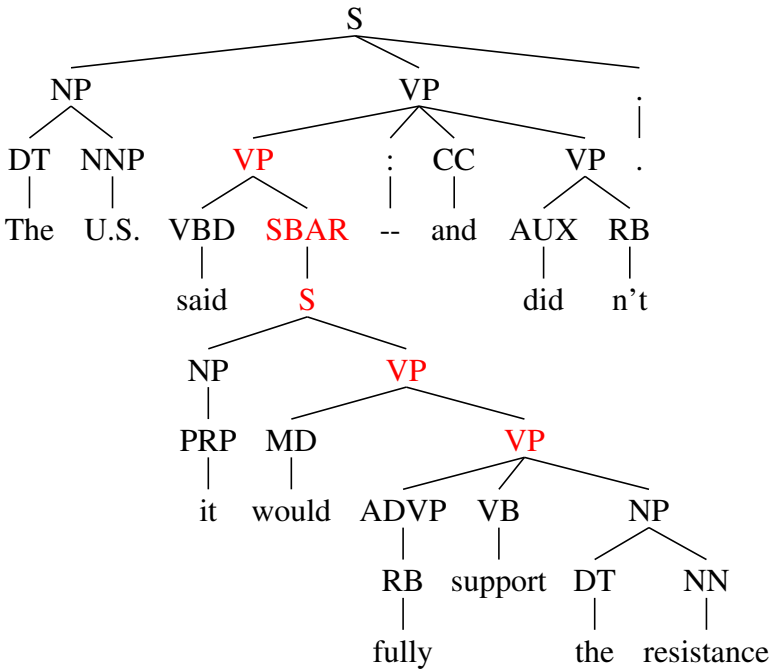

S
NP ADVP VP .
DT NN RB VBD NP PP , S .
The company also pleased NNS IN S , VP
analysts by VP VBG NP
VBG NP ending JJ NNP
announcing NP VP next August
CD JJ NN NNS VBN PP
four new store openings planned IN NP
for JJ CD
fiscal 1990


S
NP ADVP VP .
DT NN RB VBD NP PP .
The company also pleased NNS IN S
analysts by VP
VBG NP
announcing NP VP
CD JJ NN NNS VBN PP
four new store openings planned IN NP
for NP , VP
JJ CD , VBG NP
fiscal 1990 ending JJ NNP
next August


S
CC
NP
ADVP
VP
.
But
NNS
RB
AUX
VP
.
funds
generally
are
ADVP
VBN
NP
RB
prepared
DT
NN
RP
better
this
time
around


S
CC
NP
ADVP
VP
.
But
NNS
RB
AUX
ADJP
ADVP
.
funds
generally
are
RBR
JJ
NP
RB
better
prepared
DT
NN
around
this
time


S
NP
VP
.
DT
NNP
VBD
SBAR
.
The
U.S.
said
S
NP
VP
PRP
MD
ADVP
VP
it
would
RB
VP
:
CC
VP
fully
VB
NP
--
and
AUX
RB
support
DT
NN
did
n’t
the
resistance


S
NP
VP
.
DT
NNP
VP
:
CC
VP
.
The
U.S.
VBD
SBAR
--
and
AUX
RB
said
S
did
n’t
NP
VP
PRP
MD
VP
it
would
ADVP
VB
NP
RB
support
DT
NN
fully
the
resistance