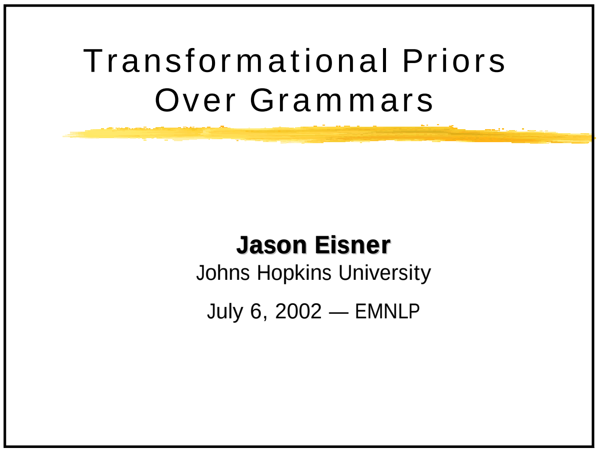
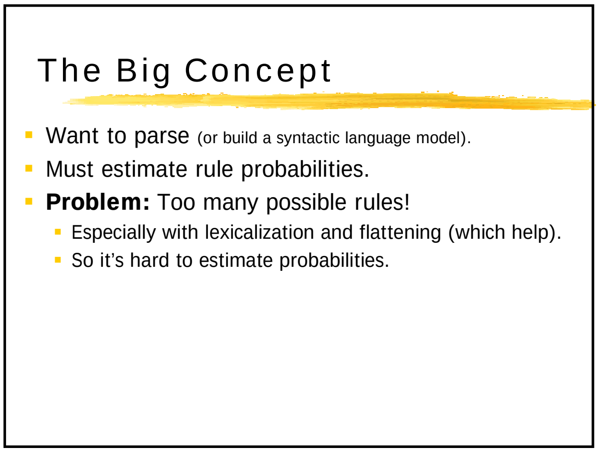


Transformational Priors Over Grammars

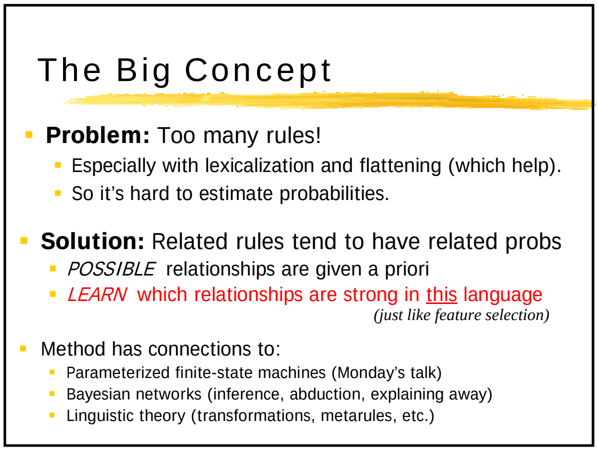
Jason Eisner
Johns Hopkins University

July 6, 2002 — EMNLP



The Big Concept

- Want to parse (or build a syntactic language model).
- Must estimate rule probabilities.
- **Problem:** Too many possible rules!
  - Especially with lexicalization and flattening (which help).
  - So it's hard to estimate probabilities.

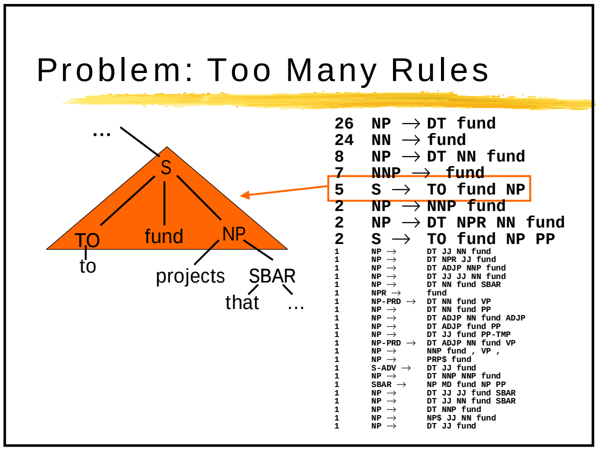


The Big Concept

- **Problem:** Too many rules!
  - Especially with lexicalization and flattening (which help).
  - So it's hard to estimate probabilities.
- **Solution:** Related rules tend to have related probs
  - *POSSIBLE* relationships are given a priori
  - *LEARN* which relationships are strong in this language

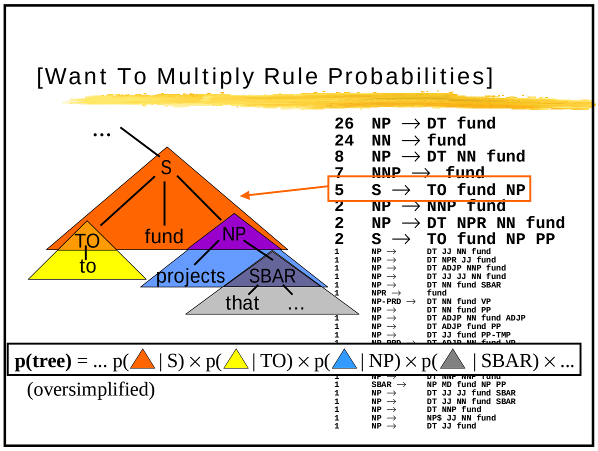
(just like feature selection)

- Method has connections to:
  - Parameterized finite-state machines (Monday's talk)
  - Bayesian networks (inference, abduction, explaining away)
  - Linguistic theory (transformations, metarules, etc.)



Problem: Too Many Rules

26	NP →	DT fund
24	NN →	fund
8	NP →	DT NN fund
7	NNP →	fund
5	S →	TO fund NP
2	NP →	NNP fund
2	NP →	DT NPR NN fund
2	S →	TO fund NP PP
1	NP →	DT JJ NN fund
1	NP →	DT NPR JJ fund
1	NP →	DT ADJP NNP fund
1	NP →	DT JJ JJ NN fund
1	NP →	DT NN fund SBAR
1	NPR →	fund
1	NP-PRD →	DT NN fund VP
1	NP →	DT NN fund PP
1	NP →	DT ADJP NN fund ADJP
1	NP →	DT ADJP fund PP
1	NP →	DT JJ fund PP-TMP
1	NP-PRD →	DT ADJP NN fund VP
1	NP →	NNP fund , VP ,
1	NP →	PRP$ fund
1	S-ADV →	DT JJ fund
1	NP →	DT NNP NNP fund
1	SBAR →	NP MD fund NP PP
1	NP →	DT JJ JJ fund SBAR
1	NP →	DT JJ NN fund SBAR
1	NP →	DT NNP fund
1	NP →	NP$ JJ NN fund
1	NP →	DT JJ fund

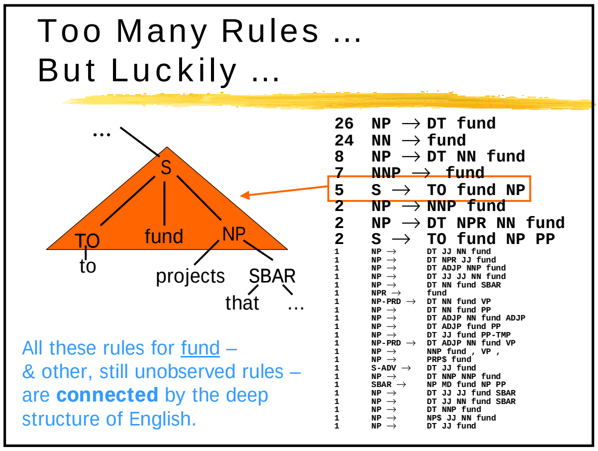


[Want To Multiply Rule Probabilities]

26	NP →	DT fund
24	NN →	fund
8	NP →	DT NN fund
7	NNP →	fund
5	S →	TO fund NP
2	NP →	NNP fund
2	NP →	DT NPR NN fund
2	S →	TO fund NP PP
1	NP →	DT JJ NN fund
1	NP →	DT NPR JJ fund
1	NP →	DT ADJP NNP fund
1	NP →	DT JJ JJ NN fund
1	NP →	DT NN fund SBAR
1	NPR →	fund
	NP-PRD →	DT NN fund VP
	NP →	DT NN fund PP
1	NP →	DT ADJP NN fund ADJP
1	NP →	DT ADJP fund PP
1	NP →	DT JJ fund PP-TMP
1	NP-PRD →	DT ADJP NN fund VP
1	NP →	DT NNP NNP fund
1	SBAR →	NP MD fund NP PP
1	NP →	DT JJ JJ fund SBAR
1	NP →	DT JJ NN fund SBAR
1	NP →	DT NNP fund
1	NP →	NP$ JJ NN fund
1	NP →	DT JJ fund

p(tree) = ... p(▲ | S) × p(▲ | TO) × p(▲ | NP) × p(▲ | SBAR) × ...

(oversimplified)



Too Many Rules ... But Luckily ...

26	NP →	DT fund
24	NN →	fund
8	NP →	DT NN fund
7	NNP →	fund
5	S →	TO fund NP
2	NP →	NNP fund
2	NP →	DT NPR NN fund
2	S →	TO fund NP PP
1	NP →	DT JJ NN fund
1	NP →	DT NPR JJ fund
1	NP →	DT ADJP NNP fund
1	NP →	DT JJ JJ NN fund
1	NP →	DT NN fund SBAR
1	NPR →	fund
1	NP-PRD →	DT NN fund VP
1	NP →	DT NN fund PP
1	NP →	DT ADJP NN fund ADJP
1	NP →	DT ADJP fund PP
1	NP →	DT JJ fund PP-TMP
1	NP-PRD →	DT ADJP NN fund VP
1	NP →	NNP fund , VP ,
1	NP →	PRP$ fund
1	S-ADV →	DT JJ fund
1	NP →	DT NNP NNP fund
1	SBAR →	NP MD fund NP PP
1	NP →	DT JJ JJ fund SBAR
1	NP →	DT JJ NN fund SBAR
1	NP →	DT NNP fund
1	NP →	NP$ JJ NN fund
1	NP →	DT JJ fund

All these rules for fund – & other, still unobserved rules – are **connected** by the deep structure of English.

Rules Are Related

- fund behaves like a typical singular noun ...

```
26  NP  → DT fund
24  NN  → fund
8   NP  → DT NN fund
7   NNP →  fund
5   S  →  TO fund NP
```

one fact!
though PCFG represents it as many apparently unrelated rules.

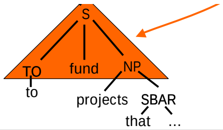
```
1   NP →      DT NPR JJ fund
1   NP →      DT ADJP NNP fund
1   NP →      DT JJ JJ NN fund
1   NP →      DT NN fund SBAR
1   NPR →     fund
1   NP-PRD →  DT NN fund VP
1   NP →      DT NN fund PP
1   NP →      DT ADJP NN fund ADJP
1   NP →      DT ADJP fund PP
1   NP →      DT JJ fund PP-TMP
1   NP-PRD →  DT ADJP NN fund VP
1   NP →      NNP fund , VP ,
1   NP →      PRP$ fund
1   S-ADV →   DT JJ fund
1   NP →      DT NNP NNP fund
1   SBAR →    NP MD fund NP PP
1   NP →      DT JJ JJ fund SBAR
1   NP →      DT JJ NN fund SBAR
1   NP →      DT NNP fund
1   NP →      NP$ JJ NN fund
1   NP →      DT JJ fund
```

Rules Are Related

- fund behaves like a typical singular noun ...
- ... **or** transitive verb ...

```
26  NP  → DT fund
24  NN  → fund
8   NP  → DT NN fund
7   NNP →  fund
5   S  →  TO fund NP
2   NP  → NNP fund
2   NP  → DT NPR NN fund
2   S  →  TO fund NP PP
1   NP →      DT JJ NN fund
1   NP →      DT NPR JJ fund
1   NP →      DT ADJP NNP fund
1   NP →      DT JJ JJ NN fund
1   NP →      DT NN fund SBAR
1   NP →      DT ADJP fund PP
1   NP →      DT JJ fund PP-TMP
1   NP-PRD →  DT ADJP NN fund VP
1   NP →      NNP fund , VP ,
1   NP →      PRP$ fund
1   S-ADV →   DT JJ fund
1   NP →      DT NNP NNP fund
1   SBAR →  NP MD fund NP PP
1   NP →      DT JJ JJ fund SBAR
1   NP →      DT JJ NN fund SBAR
1   NP →      DT NNP fund
1   NP →      NP$ JJ NN fund
1   NP →      DT JJ fund
```

one more fact!
even if several more rules.
Verb rules are **RELATED**.
Should be able to **PREDICT** the ones we haven't seen.



Rules Are Related

- fund behaves like a typical singular noun ...
- ... **or** transitive verb ...
- ... but as noun, has an idiosyncratic fondness for purpose clauses ...

```
26  NP  → DT fund
24  NN  → fund
8   NP  → DT NN fund
7   NNP →  fund
5   S  →  TO fund NP
2   NP  → NNP fund
2   NP  → DT NPR NN fund
2   S  →  TO fund NP PP
1   NP →      DT JJ NN fund
1   NP →      DT NPR JJ fund
1   NP →      DT ADJP NNP fund
1   NP →      DT JJ JJ NN fund
1   NP →      DT NN fund SBAR
1   NPR →     fund
1  NP-PRD → DT NN fund VP
1   NP →      DT NN fund PP
1   NP →      DT ADJP NN fund ADJP
1   NP →      DT ADJP fund PP
1   NP →      DT JJ fund PP-TMP
1  NP-PRD→DT ADJP NN fund VP
1  NP →    NNP fund , VP ,
    NP →      PRP$ fund
    S-ADV →   DT JJ fund
    NP →      DT NNP NNP fund
1   SBAR →    NP MD fund NP PP
1  NP →    DT JJ JJ fund SBAR
1  NP →    DT JJ NN fund SBAR
1   NP →      DT NNP fund
1   NP →      NP$ JJ NN fund
```

the ACL fund to put proceedings online
the old ACL fund for students to attend ACL
one more fact!
predicts dozens of unseen rules

Rules Are Related

- fund behaves like a typical singular noun ...
- ... **or** transitive verb ...
- ... but as noun, has an idiosyncratic fondness for purpose clauses ...
- ... and maybe other idiosyncrasies to be discovered, like unaccusativity ...

```
26  NP  → DT fund
24  NN  → fund
8   NP  → DT NN fund
7   NNP →  fund
5   S  →  TO fund NP
2   NP  → NNP fund
2   NP  → DT NPR NN fund
```

NSF issued the grant
The grant issued today
⬇ ???
NSF funded the grant
The grant funded today

unlikely sentence, but if we do see it,
is unaccusativity **plausible**? (vs. other parse)

All This Is Quantitative!

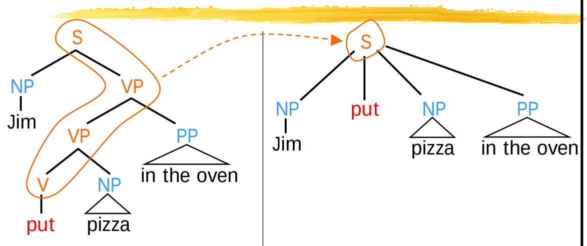
- fund behaves like a typical singular noun ...
- ... **or** transitive verb ...
- ... but as noun, has an idiosyncratic fondness for purpose clauses ...
- ... and maybe other idiosyncrasies to be discovered, like unaccusativity ...

```
26  NP  → DT fund
24  NN  → fund
8   NP  → DT NN fund
7   NNP →  fund
5   S  →  TO fund NP
2   NP  → NNP fund
2   NP  → DT NPR NN fund
```

how often?

and how does that tell us p(rule)?

Format of the Rules



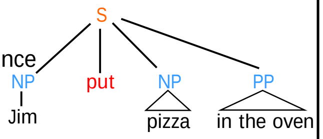
```
S   → NP VP  (put)
VP  → VP PP  (put)
VP  → V  NP  (put)
V   → put    (put)
```

```
S → NP put NP PP
```

Format of the Rules

Why use flat rules?

- Avoids silly independence assumptions: a win
  - Johnson 1998 →
  - New experiments
- Our method likes them
  - Traditional rules aren't systematically related
  - But relationships exist among wide, flat rules that express different ways of filling same roles

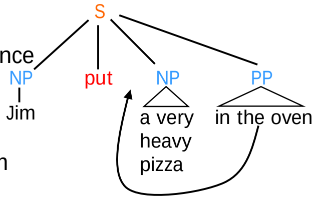


S → **NP put NP PP**

Format of the Rules

Why use flat rules?

- Avoids silly independence assumptions: a win
  - Johnson 1998 →
  - New experiments
- Our method likes them
  - Traditional rules aren't systematically related
  - But relationships exist among wide, flat rules that express different ways of filling same roles

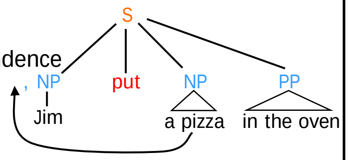


S → **NP put PP NP**

Format of the Rules

Why use flat rules?

- Avoids silly independence assumptions: a win
  - Johnson 1998 →
  - New experiments
- Our method likes them
  - Traditional rules aren't systematically related
  - But relationships exist among wide, flat rules that express different ways of filling same roles

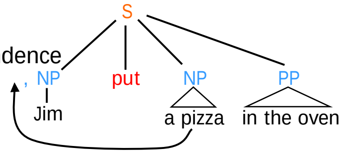


S → **NP , NP put PP**

Format of the Rules

Why use flat rules?

- Avoids silly independence assumptions: a win
  - Johnson 1998 →
  - New experiments
- Our method likes them
  - Traditional rules aren't systematically related
  - But relationships exist among wide, flat rules that express different ways of filling same roles



in short, flat rules are the **locus of transformations**

Format of the Rules

Why use flat rules?

- Avoids silly indep. assumptions: a win
  - Johnson 1998 →
  - New experiments
- Our method likes them
  - Traditional rules aren't systematically related
  - But relationships exist among wide, flat rules that express different ways of filling same roles

flat rules are the **locus of exceptions**
(e.g., put is exceptionally likely to take a PP, but not a second PP)

in short, flat rules are the **locus of transformations**

Intuition: Listing is costly and hard to learn.
Most rules are derived.

Hey – Just Like Linguistics!

Lexicalized syntactic formalisms: CG, LFG, TAG, HPSG, LCFG ...

- Grammar = set of "lexical entries" very like flat rules
- Exceptional entries OK

flat rules are the **locus of exceptions**
(e.g., put is exceptionally likely to take a PP, but not a second PP)

listed entries

derived entries

- Explain "coincidental" patterns of lexical entries: metarules/ transformations/lexical redundancy rules

in short, flat rules are the **locus of transformations**

The Rule Smoothing Task

- **Input:** Rule counts (from parses or putative parses)
- **Output:** Probability distribution over rules
- **Evaluation:** Perplexity of held-out rule counts
  - That is, did we assign high probability to the rules needed to correctly parse test data?

The Rule Smoothing Task

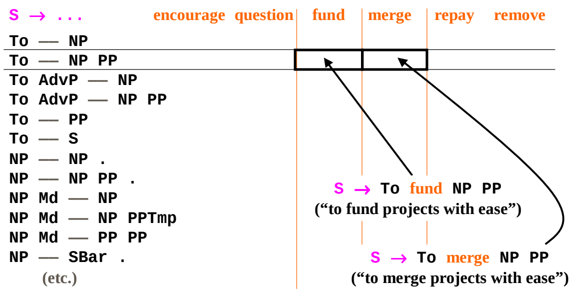
- **Input:** Rule counts (from parses or putative parses)
- **Output:** Probability distribution over rules
- **Evaluation:** Perplexity of held-out rule counts

Rule probabilities: p(S→ NP put NP PP | S,put)

Infinite set of possible rules; so we will estimate

p(S→ NP Adv PP put PP PP NP AdjP S | S, put)
= a very tiny number > 0

Grid of Lexicalized Rules



S → ... encourage question fund merge repay remove

To —— NP
To —— NP PP
To AdvP —— NP
To AdvP —— NP PP
To —— PP
To —— S
NP —— NP .
NP —— NP PP .
NP Md —— NP
NP Md —— NP PPTmp
NP Md —— PP PP
NP —— SBar .
(etc.)

S → To fund NP PP
(“to fund projects with ease”)

S → To merge NP PP
(“to merge projects with ease”)

Training Counts

S → ...	encourage	question	fund	merge	repay	remove
To —— NP	1	1	5	1	3	2
To —— NP PP	1	1	2	2	1	1
To AdvP —— NP						1
To AdvP —— NP PP						1
NP —— NP .		2				
NP —— NP PP .	1					
NP Md —— NP	1					
NP Md —— NP PPTmp					1	
NP Md —— PP PP						1
To —— PP				1		
To —— S	1					
NP —— SBar .		2				
(other)						

Count of (word, frame)

Naive prob. estimates (MLE model)

S → ...	encourage	question	fund	merge	repay	remove
To —— NP	200	167	714	250	600	333
To —— NP PP	200	167	286	500	200	167
To AdvP —— NP	0	0	0	0	0	167
To AdvP —— NP PP	0	0	0	0	0	167
NP —— NP .	0	333	0	0	0	0
NP —— NP PP .	200	0	0	0	0	0
NP Md —— NP	200	0	0	0	0	0
NP Md —— NP PPTmp	0	0	0	0	200	0
NP Md —— PP PP	0	0	0	0	0	167
To —— PP	0	0	0	250	0	0
To —— S	200	0	0	0	0	0
NP —— SBar .	0	333	0	0	0	0
(other)	0	0	0	0	0	0

Estimate of p(frame | word) * 1000

TASK: counts → probs (“smoothing”)

S → ...	encourage	question	fund	merge	repay	remove
To —— NP	142	117	397	210	329	222
To —— NP PP	77	64	120	181	88	80
To AdvP —— NP	0.55	0.47	1.1	0.82	0.91	79
To AdvP —— NP PP	0.18	0.15	0.33	0.37	0.26	50
NP —— NP .	22	161	7.8	7.5	7.9	7.5
NP —— NP PP .	79	8.5	2.6	2.7	2.6	2.6
NP Md —— NP	90	2.1	2.4	2.0	24	2.6
NP Md —— NP PPTmp	1.8	0.16	0.17	0.16	69	0.19
NP Md —— PP PP	0.1	0.027	0.027	0.038	0.078	59
To —— PP	9.2	6.5	12	126	10	9.1
To —— S	98	1.6	4.3	3.9	3.6	2.7
NP —— SBar .	3.4	190	3.2	3.2	3.2	3.2
(other)	478	449	449	461	461	482

Estimate of p(frame | word) * 1000

Smooth Matrix via LSA / SVD, or SBS?

S → ...	encourage	question	fund	merge	repay	remove
To —— NP	1	1	5	1	3	2
To —— NP PP	1	1	2	2	1	1
To AdvP —— NP						1
To AdvP —— NP PP						1
NP —— NP .		2				
NP —— NP PP .	1					
NP Md —— NP	1					
NP Md —— NP PPTmp					1	
NP Md —— PP PP						1
To —— PP				1		
To —— S	1					
NP —— SBar .		2				
(other)						

Count of (word, frame)

Smoothing via a Bayesian Prior

- Choose grammar to maximize
  p(observed rule counts | grammar)*p(grammar)
- grammar = probability distribution over rules
- **Our job:** Define p(grammar)
- **Question:** What makes a grammar likely, a priori?
- **This paper's answer:** Systematicity.
  Rules are mainly derivable from other rules.
  Relatively few stipulations ("deep facts").

Only a Few Deep Facts

- fund behaves like a transitive verb 10% of time ...
- and noun 90% of time ...
- ... takes purpose clauses 5 times as often as typical noun.

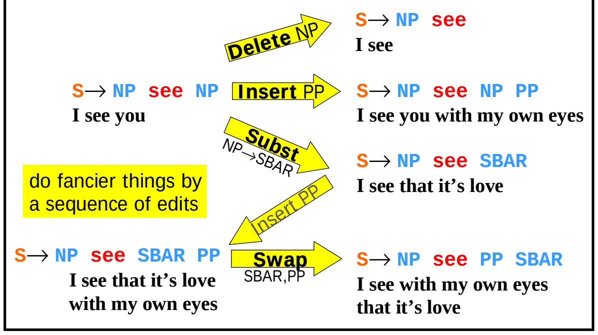
26	NP →	DT fund
24	NN →	fund
8	NP →	DT NN fund
7	NNP →	fund
5	S →	TO fund NP
2	NP →	NNP fund
2	NP →	DT NPR NN fund
2	S →	TO fund NP PP
1	NP →	DT JJ NN fund
1	NP →	DT NPR JJ fund
1	NP →	DT ADJP NNP fund
1	NP →	DT JJ JJ NN fund
1	NP →	DT NN fund SBAR
1	NPR →	fund
1	NP-PRD →	DT NN fund VP
1	NP →	DT NN fund PP
1	NP →	DT ADJP NN fund ADJP
1	NP →	DT ADJP fund PP
1	NP →	DT JJ fund PP-TMP
1	NP-PRD →	DT ADJP NN fund VP
1	NP →	NNP fund , VP ,
1	NP →	PRP$ fund
1	S-ADV →	DT JJ fund
1	NP →	DT NNP NNP fund
1	SBAR →	NP MD fund NP PP
1	NP →	DT JJ JJ fund SBAR
1	NP →	DT JJ NN fund SBAR
1	NP →	DT NNP fund
1	NP →	NP$ JJ NN fund
1	NP →	DT JJ fund

Smoothing via a Bayesian Prior

- Previous work (several papers in past decade):
  - Rules should be few, short, and approx. equiprobable
  - These priors try to keep rules **out of** grammar
  - Bad idea for lexicalized grammars ...
- This work:
  - Prior tries to get related rules **into** grammar
  - transitive ➔ passive at ≈1/20 the probability
    - NSF spraggles the project ➔ The project is spraggled by NSF
    - Would be weird for the passive to be missing, and prior knows it!
    - In fact, weird if p(passive) is too far from 1/20 * p(active)
  - Few facts, not few rules!

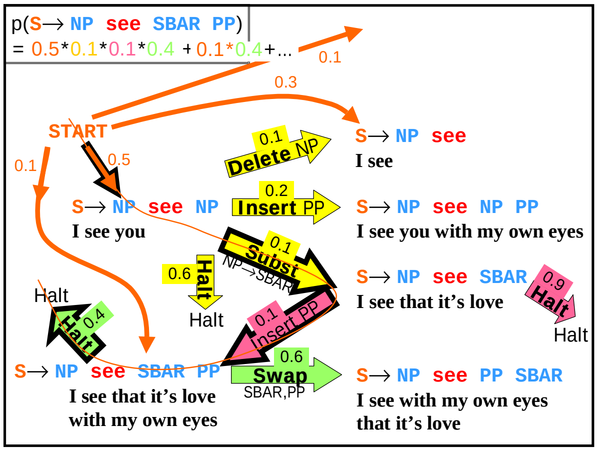
for now, stick to Simple Edit Transformations

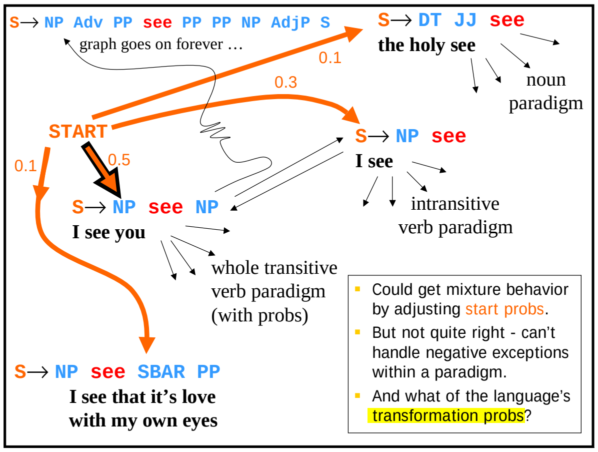
See paper for various evidence that these should be predictive.



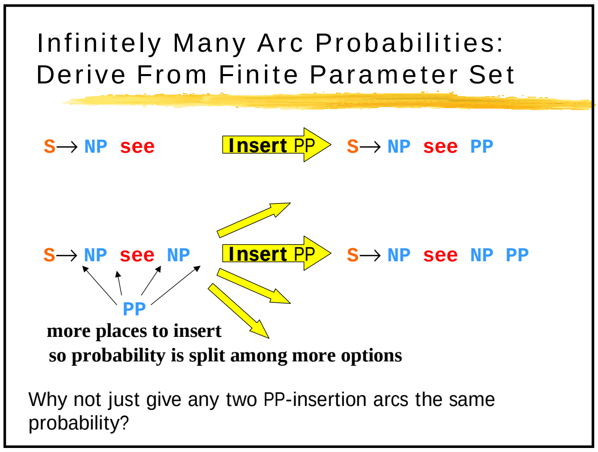
do fancier things by a sequence of edits

p(S→ NP see SBAR PP)
= 0.5*0.1*0.1*0.4 + 0.1*0.4+...





- Could get mixture behavior by adjusting start probs.
- But not quite right - can't handle negative exceptions within a paradigm.
- And what of the language's transformation probs?

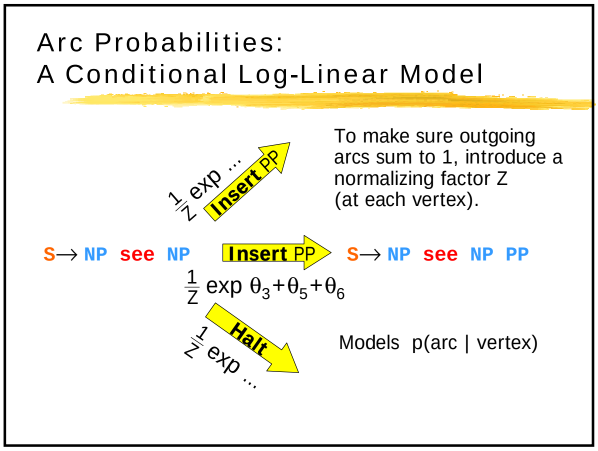


Infinitely Many Arc Probabilities: Derive From Finite Parameter Set

more places to insert

so probability is split among more options

Why not just give any two PP-insertion arcs the same probability?

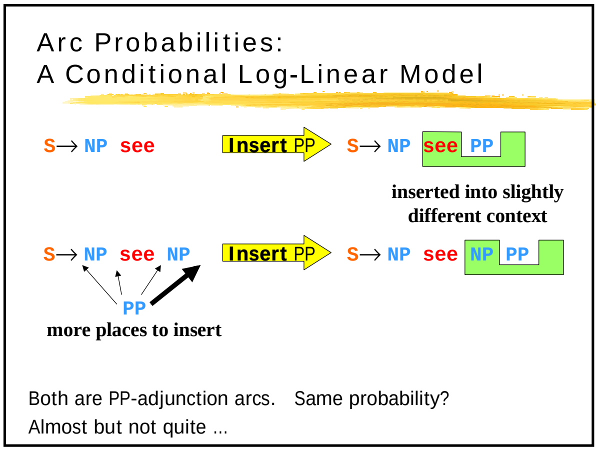


Arc Probabilities: A Conditional Log-Linear Model

$\frac{1}{Z}$ exp $\theta_3+\theta_5+\theta_6$

To make sure outgoing arcs sum to 1, introduce a normalizing factor Z (at each vertex).

Models p(arc | vertex)



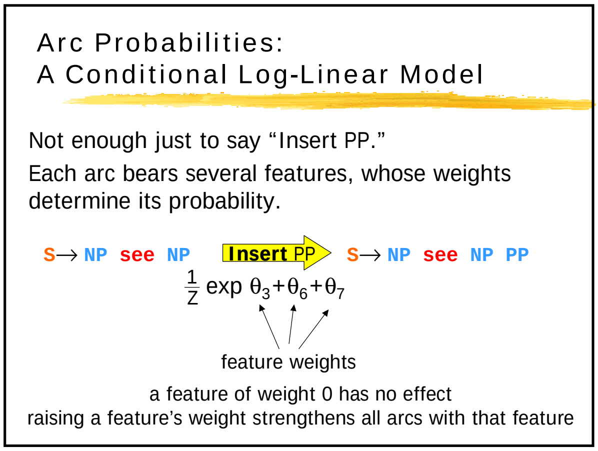
Arc Probabilities: A Conditional Log-Linear Model

inserted into slightly different context

more places to insert

Both are PP-adjunction arcs. Same probability?

Almost but not quite …



Arc Probabilities: A Conditional Log-Linear Model

Not enough just to say "Insert PP."

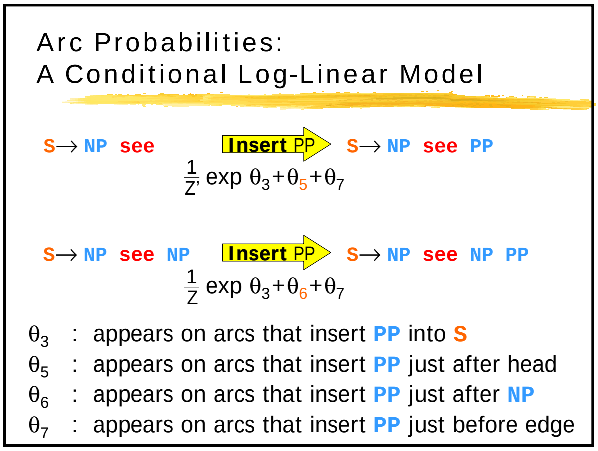
Each arc bears several features, whose weights determine its probability.

$\frac{1}{Z}$ exp $\theta_3+\theta_6+\theta_7$

feature weights

a feature of weight 0 has no effect

raising a feature's weight strengthens all arcs with that feature



Arc Probabilities: A Conditional Log-Linear Model

$\frac{1}{Z'}$ exp $\theta_3+\theta_5+\theta_7$

$\frac{1}{Z}$ exp $\theta_3+\theta_6+\theta_7$

θ_3 : appears on arcs that insert PP into S

θ_5 : appears on arcs that insert PP just after head

θ_6 : appears on arcs that insert PP just after NP

θ_7 : appears on arcs that insert PP just before edge

Arc Probabilities: A Conditional Log-Linear Model

S→ NP see **Insert** PP S→ NP see PP

$\frac{1}{Z'}\exp$ θ_3 $+\theta_5$ $+\theta_7$

S→ NP see NP **Insert** PP S→ NP see NP PP

$\frac{1}{Z}\exp$ θ_3 $+\theta_6$ $+\theta_7$

θ_3 : appears on arcs that insert **PP** into **S**

θ_5 : appears on arcs that insert **PP** just after head

θ_6 : appears on arcs that insert **PP** just after **NP**

θ_7 : appears on arcs that insert **PP** just before edge

Arc Probabilities: A Conditional Log-Linear Model

S→ NP see **Insert** PP S→ NP see PP

$\frac{1}{Z'}\exp$ θ_3 $+\theta_5$ $+\theta_7$

S→ NP see NP **Insert** PP S→ NP see NP PP

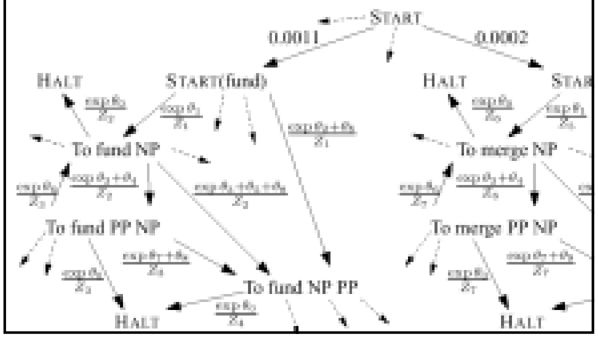
$\frac{1}{Z}\exp$ θ_3 $+\theta_6$ $+\theta_7$

These arcs share most features.
So their probabilities tend to rise and fall together.
To fit data, could manipulate them independently (via θ_5,θ_6).

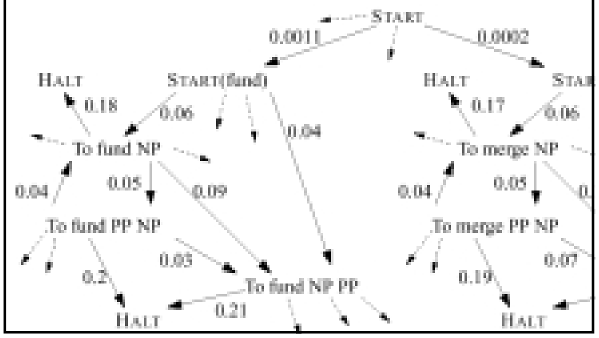
Prior Distribution

- PCFG grammar is **determined** by θ_0, θ_1, θ_2, ...

Universal Grammar



Instantiated Grammar



Prior Distribution

- Grammar is **determined** by θ_0, θ_1, θ_2, ...
- Our prior: $\theta_i \sim N(0, \sigma^2)$, IID
- Thus: -log p(grammar) = c+ $(\theta_0^2+\theta_1^2+\theta_2^2+\ldots)/\sigma^2$

- So good grammars have few large weights.
- Prior prefers one generalization to many exceptions.

Arc Probabilities: A Conditional Log-Linear Model

S→ NP see   Insert PP   S→ NP see PP

$\frac{1}{Z'}\exp$ θ_3 $+\theta_5$ $+\theta_7$

S→ NP see NP   Insert PP   S→ NP see NP PP

$\frac{1}{Z}\exp$ θ_3 $+\theta_6$ $+\theta_7$

To raise both rules' probs, cheaper to use θ_3 than both θ_5 & θ_6.
This generalizes – also raises other cases of PP-insertion!

Arc Probabilities: A Conditional Log-Linear Model

S→ NP fund NP   Insert PP   S→ NP fund NP PP

$\frac{1}{Z''}\exp$ θ_3 $+\theta_{82}$ $+\theta_6$ $+\theta_7$

S→ NP see NP   Insert PP   S→ NP see NP PP

$\frac{1}{Z}\exp$ θ_3 $+\theta_{84}$ $+\theta_6$ $+\theta_7$

To raise both probs, cheaper to use θ_3 than both θ_{82} & θ_{84}.
This generalizes – also raises other cases of PP-insertion!

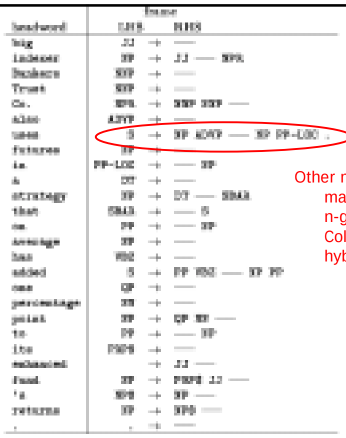
Reparameterization

- Grammar is determined by θ_0, θ_1, θ_2, …
- A priori, the θ_i are normally distributed

- We've reparameterized!
- The parameters are feature weights θ_i, not rule probabilities
- Important tendencies captured in big weights
  - Similarly: Fourier transform – find the formants
  - Similarly: SVD – find the principal components
  - It's on this deep level that we want to compare events, impose priors, etc.

```
( (S
    (NP-SBJ (JJ Big) (NN indexer) (NNP Bankers) (NNP Trust) (NNP Co.) )
    (ADVP (RB also) )
    (VP (VBZ uses)
      (NP (NNS futures) )
      (PP-LOC (IN in)
        (NP
          (NP (DT a) (NN strategy) )
          (SBAR
            (WHNP-1 (IN that) )
            (S
              (PP (IN on)
                (NP (NN average) ))
              (NP-SBJ (-NONE- *T*-1) )
              (VP (VBZ has)
                (VP (VBN added)
                  (NP (CD one) (NN percentage) (NN point) )
                  (PP-CLR (TO to)
                    (NP
                      (NP (PRP$ its) (JJ enhanced) (NN fund) (POS 's) )
                      (NNS returns) ))))))))))
    (. .) ))
```

```
( (S
    (NP (JJ big) indexer (NPR (NNP Bankers) (NNP Trust) Co.) )
    (ADVP also)
    uses
    (NP futures)
    (PP-LOC in
      (NP
        (NP (DT a)
            strategy
            (SBAR that
              (S
                (PP on (NP average) )
                (VBZ has)
                added
                (NP (CD one) (NN percentage) point)
                (PP to
                  (NP
                    (NPB (NP (PRP$ its) (JJ enhanced) fund) 's)
                    returns) ))))))
    (. .) ))
```

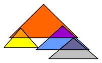


headword	frame		
	LHS		RHS
big	JJ	→	—
indexer	NP	→	JJ — NPR
Bankers	NNP	→	—
Trust	NNP	→	—
Co.	NPR	→	NNP NNP —
also	ADVP	→	—
uses	S	→	NP ADVP — NP PP-LOC .
futures	NP	→	—
in	PP-LOC	→	— NP
a	DT	→	—
strategy	NP	→	DT — SBAR
that	SBAR	→	— S
on	PP	→	— NP
average	NP	→	—
has	VBZ	→	—
added	S	→	PP VBZ — NP PP
one	CD	→	—
percentage	NN	→	—
point	NP	→	CD NN —
to	PP	→	— NP
its	PRP$	→	—
enhanced		→	JJ —
fund	NP	→	PRP$ JJ —
's	NPB	→	NP —
returns	NP	→	NPB —
.	.	→	—

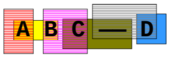
Other models of this string:
max-likelihood
n-gram
Collins arg/adj
hybrids

Simple Bigram Model (Eisner 1996)

- A parser assumes tree is probable if its component rules are:



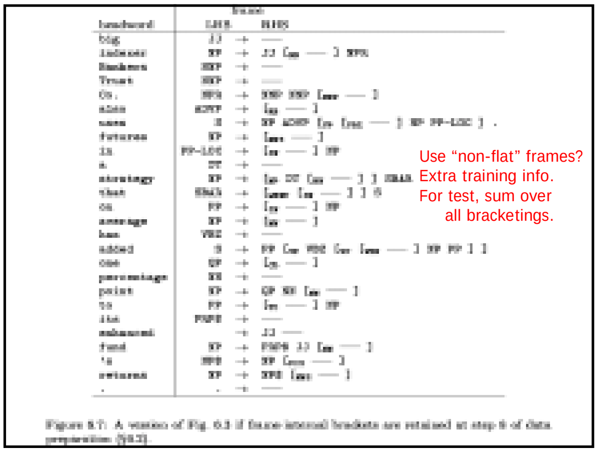
- Try assuming rule is probable if its component bigrams are:



p(A | start) × p(B | A)
× p(C | B) × p(—— | C)
× p(D | ——) × p(stop | D)

- Markov process, 1 symbol of memory; conditioned on L, w, side of ——
- One-count backoff to handle sparse data (Chen & Goodman 1996)

p(L → A B C — D | w) = p(L | w)• p(A B C — D | L,w)



headword	frame LHS		frame RHS
big	JJ	→	—
index	NP	→	[illegible]
Bankers	NNP	→	—
Trust	NNP	→	—
Co.	NPS	→	[illegible]
also	ADVP	→	[illegible]
uses	S	→	[illegible]
futures	NP	→	[illegible]
in	PP-LOC	→	[illegible]
a	DT	→	—
strategy	NP	→	[illegible]
that	SBAR	→	[illegible]
on	PP	→	[illegible]
average	NP	→	[illegible]
has	VBZ	→	—
added	S	→	[illegible]
one	QP	→	[illegible]
percentage	NN	→	—
point	NP	→	[illegible]
to	PP	→	[illegible]
its	PRP$	→	—
enhanced		→	JJ —
fund	NP	→	[illegible]
's	POS	→	[illegible]
returns	NP	→	[illegible]
.	.	→	—

Figure 8.7: A version of Fig. 6.3 if frame-internal brackets are retained at step 6 of data preparation (§6.3).

Use “non-flat” frames?
Extra training info.
For test, sum over
all bracketings.

Perplexity: Predicting test frames

	basic	
	flat	non-flat[b]
Treebank	∞	∞
1-gram	1774.9	86435.1
2-gram	135.2	199.3
3-gram	136.5	177.4
Collins[c]	363.0	494.5
transformation	108.6	

20% further
reduction

from previous lit.

Can get big perplexity reduction
just by flattening.

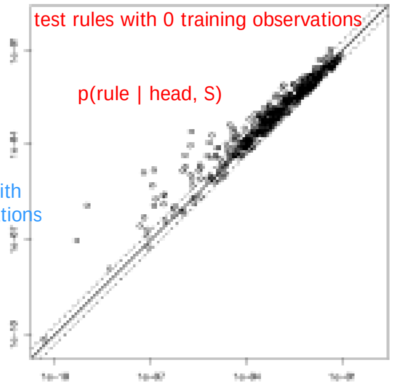
Perplexity: Predicting test frames

	basic		Treebank/Markov		
			Katz	one-count[a]	
	flat	non-flat[b]	flat	flat	non-flat
Treebank	∞	∞			
1-gram	1774.9	86435.1	340.9	160.0	193.2
2-gram	135.2	199.3	127.2	116.2	174.7
3-gram	136.5	177.4	132.7	123.5	174.8
Collins[c]	363.0	494.5	197.9		
transformation	108.6				
averaged[d]	102.3				

best model
without transformations

from previous lit

best model
with transformations

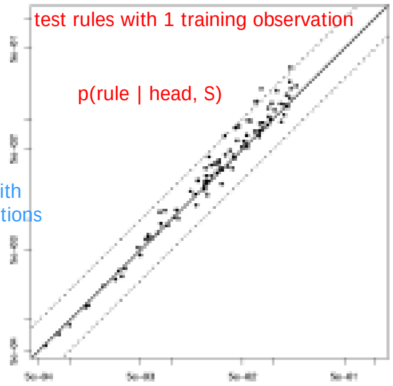


test rules with 0 training observations

p(rule | head, S)

best
model with
transformations

best model without transformations

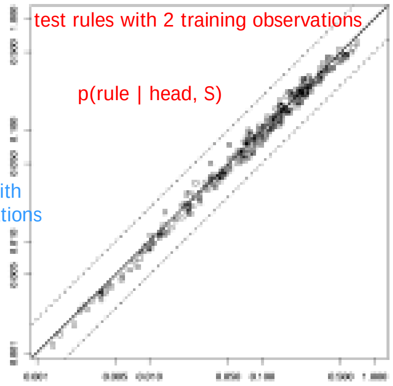


test rules with 1 training observation

p(rule | head, S)

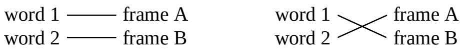
best
model with
transformations

best model without transformations



Forced matching task

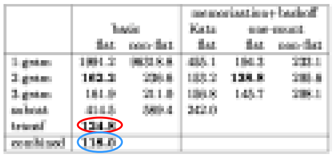
- Test model's ability to extrapolate novel frames for a word
- Randomly select **two** (word, frame) pairs from test data
  - ... ensuring that neither frame was ever seen in training
- Ask model to choose a matching:



i.e., does frame A look more like word 1's known frames or word 2's?

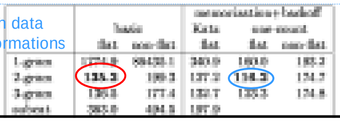
- 20% fewer errors than bigram model

Graceful degradation



	basic		memorization+backoff		
			Katz	one-count	
	flat	non-flat	flat	flat	non-flat
1-gram	1991.2	96539.6	455.1	1082.2	333.1
2-gram	**162.2**	230.6	133.2	**129.8**	208.6
3-gram	161.9	211.8	136.9	145.7	209.1
Collins	414.5	589.4	242.0		
transf.	**124.8**				
combined	118.0				

Twice as much data
But no transformations



	basic		memorization+backoff		
			Katz	one-count	
	flat	non-flat	flat	flat	non-flat
1-gram	1774.9	86435.1	340.9	160.0	193.2
2-gram	**135.2**	199.3	127.2	**118.3**	174.7
3-gram	136.5	177.4	132.7	133.3	174.8
Collins	363.0	494.5	197.9		

Summary: Reparameterize PCFG in terms of deep transformation weights, to be learned under a simple prior.

- **Problem:** Too many rules!
  - Especially with lexicalization and flattening (which help).
  - So it's hard to estimate probabilities.
- **Solution:** Related rules tend to have related probs
  - *POSSIBLE* relationships are given a priori
  - *LEARN* which relationships are strong in this language

(just like feature selection)

- Method has connections to:
  - Parameterized finite-state machines (Monday's talk)
  - Bayesian networks (inference, abduction, explaining away)
  - Linguistic theory (transformations, metarules, etc.)

FIN