Non-Parametric Bayesian Models for Natural Language

Sharon Goldwater, Tom Griffiths and Mark Johnson

Brown University

March 2006

Outline

Research goals

- A Bayesian model of language acquisition

  Input: Child-directed speech and its non-linguistic context

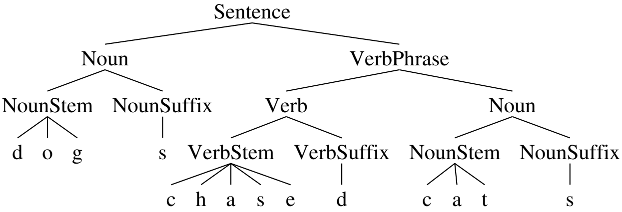
  Output: A grammar and linguistic analyses
- What information is available in different components of the input?
  - Non-linguistic context
  - Prosody
  - Transitional probabilities between phonemes or syllables
- How useful is prior knowledge, i.e. *linguistic universals*?
- Are there *synergies in language acquisition*?
  - learn syntax better if semantics learnt at same time?
  - learn lexicon better if phonology/morphology learnt at same time?

Research strategy

- Start with phonology, morphology and lexicon;
  leave syntax and semantics until later
  - children learn (some) words and inflections before they learn what they mean
  - child-directed speech corpora are readily available;
    contextual information is not
- Goal of this research (as yet unachieved):

  Input: "d o g s c h a s e d c a t s"
  (we actually use broad phonemic transcription)

  Output:



Talk summary

- Overdispersion $\Rightarrow$ PCFGs are poor models of linguistic structure
- Estimating from *types* instead of *tokens* reduces overdispersion
  ...but is only possible in simple cases
- Pitman-Yor processes provide systematic way of downsampling tokens to types (or something in between)
- Define probability distribution over CFG trees by *associating each nonterminal with its own Pitman-Yor process*
  - CFG defines *possible structures*
  - Pitman-Yor process defines *probability of each (sub)structure*
- MCMC algorithms sample posterior tree distribution given strings
- Grammars based on PY processes recover linguistic structure where ML estimation of PCFGs fail

Outline

Context-free grammar

A *context-free grammar* $G = (T, N, S, R)$ consists of:

- a finite set of *terminal symbols* T,
- a finite set of *nonterminal symbols* N disjoint from T,
- a *start symbol* $S \in N$, and
- a finite set R of *productions* $A \rightarrow \beta$ where $A \in N$ and $\beta \in (N \cup T)^+$

G *generates* a finite, labeled, ordered tree t iff:

- the *root node* of t is labeled S,
- the label of every *leaf node* of t is a member of T
- if a non-leaf node in t is labeled A and the sequence of its children's labels is β, then $A \rightarrow \beta \in R$

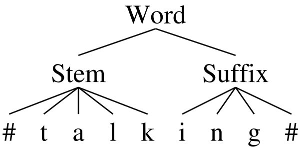
G generates a string $w \in T^+$ iff G generates a tree whose *yield* is w

Context-free grammar example

Let $G_0 = (T_0, N_0, \text{Word}, R_0)$ where:

- $T_0 = \{\text{a}, \ldots, \text{z}, \#\}$,
- $N_0 = \{\text{Stem}, \text{Suffix}, \text{Word}\}$,
- $R_0 = \left\{ \begin{array}{l} \text{Word} \to \text{Stem Suffix}, \text{Stem} \to \#\,\text{t a l k}, \\ \text{Stem} \to \#\,\text{j u m p}, \text{Suffix} \to \#, \text{Suffix} \to \text{i n g}\,\# \end{array} \right\}$

Then G_0 generates the following trees:



and thus generates the strings $\#\,\text{w a l k i n g}\,\#$ and $\#\,\text{j u m p}\,\#$

Probabilistic context-free grammar

A *probabilistic context-free grammar* is a pair (G, θ) where:

- $G = (T, N, S, R)$ is a CFG, and
- $\theta = \{\theta_r : r \in R\}$ is a set of *production probabilities* indexed by R, where $\forall r \in R\ \theta_r \geq 0$ and for all $A \in N$, $\sum_{A \to \beta \in R_A} \theta_{A \to \beta} = 1$, where R_A is subset of R with lhs A.

If t is a tree generated by G,

$$\mathrm{P}(t|\theta) = \prod_{A \to \beta \in R} {\theta_{A \to \beta}}^{f_{A \to \beta}(t)}$$

where $f_{A \to \beta}(t)$ is the number of times a node labeled A appears with children labeled β in t.

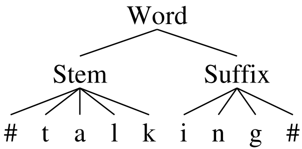
Probabilistic context-free grammar example

Let G_0 be as before, and let θ be defined as:

Production r	θ_r
Word $\rightarrow$ Stem Suffix	1.0
Stem $\rightarrow$ # t a l k	0.6
Stem $\rightarrow$ # j u m p	0.4
Suffix $\rightarrow$ #	0.7
Suffix $\rightarrow$ i n g #	0.3

Then:

$\mathrm{P}\Big($

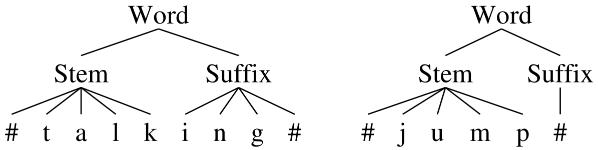


$\Big| \theta \Big) = 1.0 \times 0.6 \times 0.7$

Outline

Learning English verbal morphology

Training data is a sequence of verbs, e.g.
$\mathcal{D} = (\ \#\ \text{t a l k i n g}\ \#,\ \#\ \text{j u m p}\ \#, \ldots)$
Our goal is to infer trees such as:

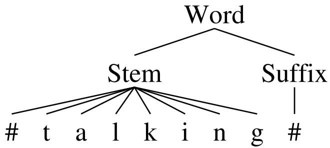


$$
\begin{array}{lll}
\text{Word} \rightarrow \text{Stem Suffix} & & \\
\text{Stem} \rightarrow w & & w \in \mathcal{T} \\
\text{Suffix} \rightarrow w & & w \in \mathcal{F}
\end{array}
$$

where $\mathcal{T}$ is the set of all prefixes of words in $\mathcal{D}$ and $\mathcal{F}$ is the set of all suffixes of words in $\mathcal{D}$

Maximum likelihood estimate for θ is trivial

- Maximum likelihood selects θ that minimizes KL-divergence between model and data distributions
- *Saturated model* with $\theta_{\mathsf{Suffix} \to \#} = 1$ generates training data distribution $\mathcal{D}$ exactly
- Saturated model is maximum likelihood estimate
- Maximum likelihood estimate does not find any suffixes



Bayesian estimation

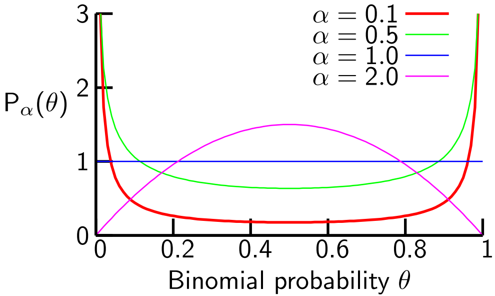
$$\underbrace{\mathrm{P(Hypothesis|Data)}}_{\text{Posterior}} \propto \underbrace{\mathrm{P(Data|Hypothesis)}}_{\text{Likelihood}} \underbrace{\mathrm{P(Hypothesis)}}_{\text{Prior}}$$

- Priors can be sensitive to linguistic structure (e.g., a word should contain a vowel)
- Priors can encode linguistic universals and markedness preferences (e.g., complex clusters appear at word onsets)
- Priors can prefer *sparse solutions*
- The choice of the prior is as much a linguistic issue as the design of the grammar!

Dirichlet priors and sparse solutions

- The probabilities $\theta_{A \to \beta}$ of choosing productions $A \to \beta$ to expand nonterminal A define multinomial distributions
- Dirichlet distributions are the *conjugate priors* to multinomials

$$\mathrm{P}(\theta_{A\to\beta_1}, \ldots, \theta_{A\to\beta_n}) \quad \propto \quad \prod_{i=1}^{n} {\theta_{A\to\beta_i}}^{\alpha-1} \qquad \alpha > 0$$



- We have developed MCMC algorithms for sampling from the posterior distribution of trees given strings $\mathcal{D}$

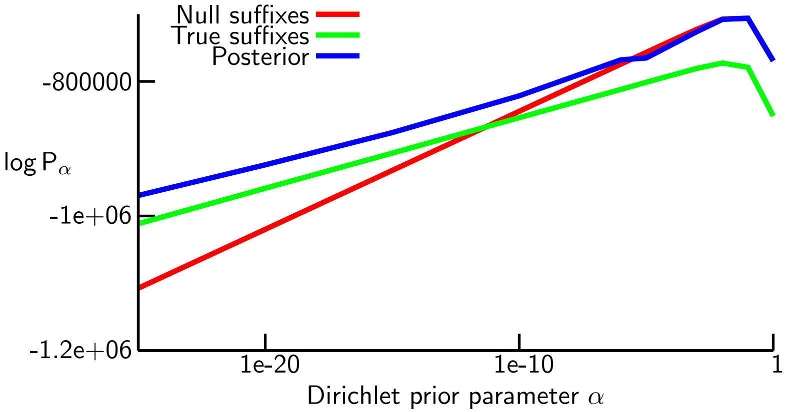
Morphological segmentation experiment

- Trained on orthographic verbs from U Penn. Wall Street Journal treebank
- Dirichlet prior prefers sparse solutions (sparser solutions as $\alpha \to 0$)
- MCMC Sampler used to sample from posterior distribution of parses
  - reanalyses each word based on a grammar estimated from the parses of the other words

Posterior samples from WSJ verb tokens

$\alpha = 0.1$	$\alpha = 10^{-5}$		$\alpha = 10^{-10}$		$\alpha = 10^{-15}$	
expect	expect		expect		expect	
expects	expects		expects		expects	
expected	expected		expected		expected	
expecting	expect	ing	expect	ing	expect	ing
include	include		include		include	
includes	includes		includ	es	includ	es
included	included		includ	ed	includ	ed
including	including		including		including	
add	add		add		add	
adds	adds		adds		add	s
added	added		add	ed	added	
adding	adding		add	ing	add	ing
continue	continue		continue		continue	
continues	continues		continue	s	continue	s
continued	continued		continu	ed	continu	ed
continuing	continuing		continu	ing	continu	ing
report	report		report		report	

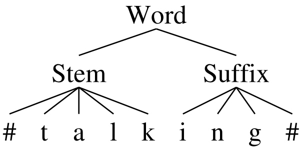
Log posterior of models on token data



- Correct solution is nowhere near as likely as posterior

$\Rightarrow$ model is wrong!

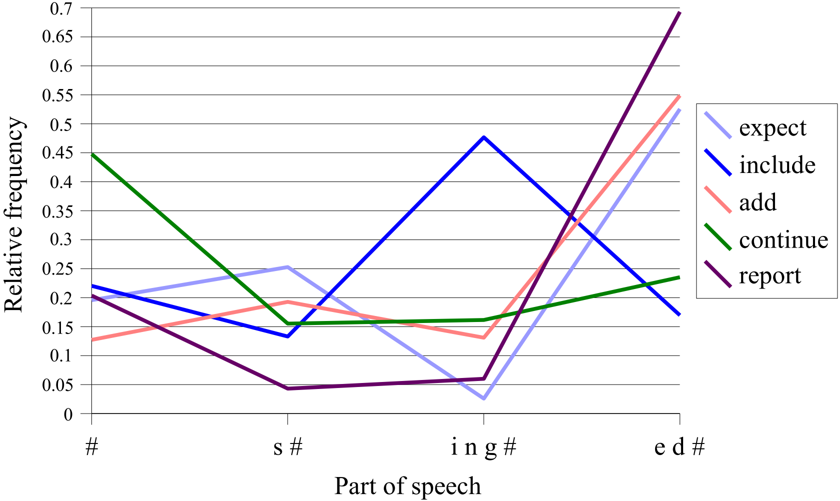
Independence assumption in PCFG model



$$P\left(\text{[tree]} \mid \theta \right) = \theta_{\mathsf{Word} \to \mathsf{Stem\ Suffix}} \, \theta_{\mathsf{Stem} \to \# \, \mathsf{t\,a\,l\,k}} \, \theta_{\mathsf{Suffix} \to \mathsf{i\,n\,g} \, \#}$$

- Model assumes relative frequency of each suffix *to be the same for all stems*
- This turns out to be incorrect

Relative frequencies of inflected verb forms



Types and tokens

- A word *type* is a distinct word shape
- A word *token* is an occurrence of a word

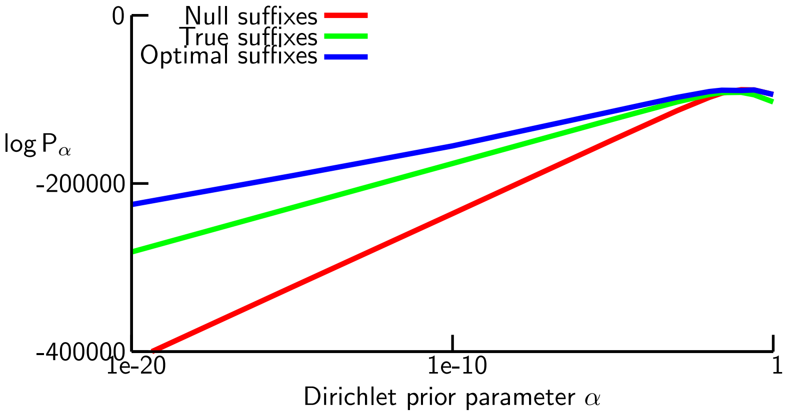
$$\begin{aligned} \text{Data} &= \text{``the cat chased the other cat''} \\ \text{Tokens} &= \text{``the''}, \text{``cat''}, \text{``chased''}, \text{``the''}, \text{``other''}, \text{``cat''} \\ \text{Types} &= \text{``the''}, \text{``cat''}, \text{``chased''}, \text{``other''} \end{aligned}$$

- Estimating θ from *word types* rather than word tokens eliminates (most) frequency variation
  - 4 common verb suffixes, so when estimating from verb types $\theta_{\text{Suffix}\rightarrow\text{i n g \#}} \approx 0.25$
- Several psycholinguists believe that humans learn morphology from word types

Posterior samples from WSJ verb *types*

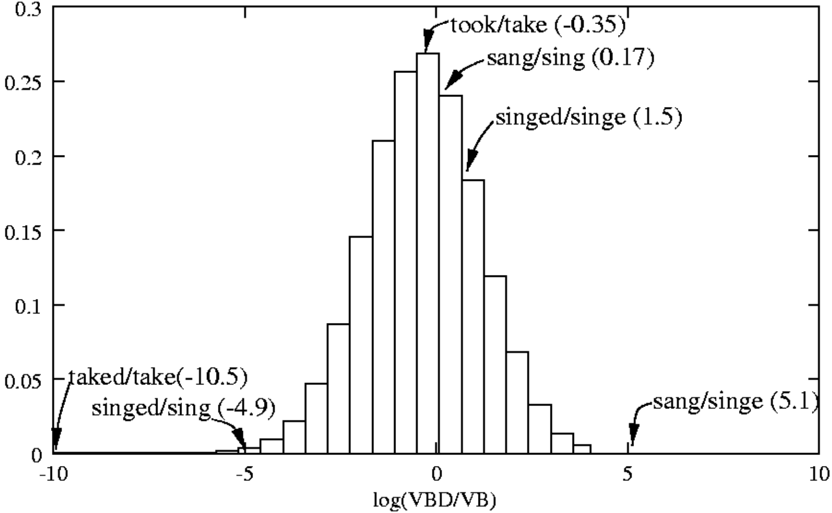
$\alpha = 0.1$		$\alpha = 10^{-5}$		$\alpha = 10^{-10}$		$\alpha = 10^{-15}$	
expect		expect		expect		exp	ect
expects		expect	s	expect	s	exp	ects
expected		expect	ed	expect	ed	exp	ected
expect	ing	expect	ing	expect	ing	exp	ecting
include		includ	e	includ	e	includ	e
include	s	includ	es	includ	es	includ	es
included		includ	ed	includ	ed	includ	ed
including		includ	ing	includ	ing	includ	ing
add		add		add		add	
adds		add	s	add	s	add	s
add	ed	add	ed	add	ed	add	ed
adding		add	ing	add	ing	add	ing
continue		continu	e	continu	e	continu	e
continue	s	continu	es	continu	es	continu	es
continu	ed	continu	ed	continu	ed	continu	ed
continuing		continu	ing	continu	ing	continu	ing
report		report		repo	rt	rep	ort

Log posterior of models on type data



- Correct solution is close to optimal at $\alpha = 10^{-3}$

Morpheme frequencies provide useful information



Yarowsky and Wicentowski (2000) "Minimally supervised Morphological Analysis by Multimodal Alignment"

Types can be hard to find

- Over-dispersion in morphological structure

⇒ PCFG estimation finds linguistic structure when training from types rather than tokens

- but speech is not segmented into words ("s e e t h e d o g g i e"), so we don't know what the types are.

⇒ integrate word segmentation with (type-based) morphology induction

- Over-dispersion occurs at virtualy all levels of linguistic structure
  - "Stocks rose" is most frequent sentence in WSJ
  - "do you", "what's that" are surprisingly frequent in child-directed speech

⇒ *type-based inference at multiple levels simultaneously*

Outline

PCFGs as recursive mixtures

The distributions over strings induced by a PCFG in *Chomsky-normal form* (i.e., all productions are of the form $A \rightarrow B\ C$ or $A \rightarrow w$, where $A, B, C \in N$ and $w \in T$) is G_S where:

$$G_A = \sum_{A \rightarrow B\ C \in R_A} \theta_{A \rightarrow B\ C} G_B \bullet G_C + \sum_{A \rightarrow w \in R_A} \theta_{A \rightarrow w} \delta_w$$

$$(P \bullet Q)(z) = \sum_{xy=z} P(x) Q(y)$$

$$\delta_w(x) = 1 \text{ if } w = x \text{ and } 0 \text{ otherwise}$$

In fact, $G_A(x) = \mathrm{P}(A \Rightarrow^\star x | \theta)$, the sum of the probability of all trees with root node A and yield x

Grammars based on Pitman-Yor processes

A Pitman-Yor grammar (G, θ, a, b) is a PCFG (G, θ) together with parameter vectors a, b where for each $A \in N$, a_A, b_A are the two parameters of a PY.

$$
\begin{aligned}
G_A &\sim \mathrm{PY}(a_A, b_A, H_A) \\
H_A &= \sum_{A \to B\ C \in R_A} \theta_{A \to B\ C} G_B \bullet G_C + \sum_{A \to w \in R_A} \theta_{A \to w} \delta_w
\end{aligned}
$$

The probabilistic language defined by the grammar is G_S.
There is one Pitman-Yor process $\mathrm{PY}(\alpha_A, H_A)$ for each nonterminal A.
Its base distribution H_A is a mixture of the Pitman-Yor processes for other nonterminals.

- Q: For what (G, θ, a, b) do these distributions exist?

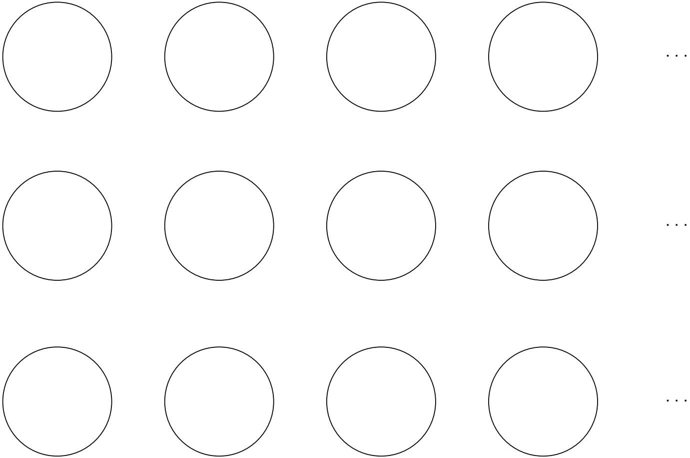
Outline

Restaurant metaphor (0)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



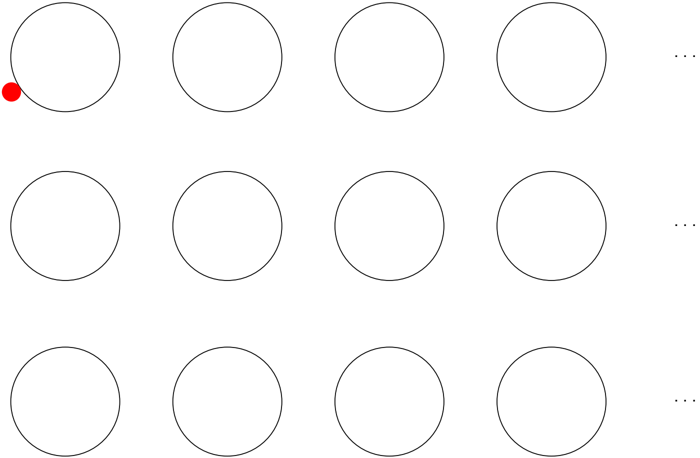
Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

Restaurant metaphor (1a)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



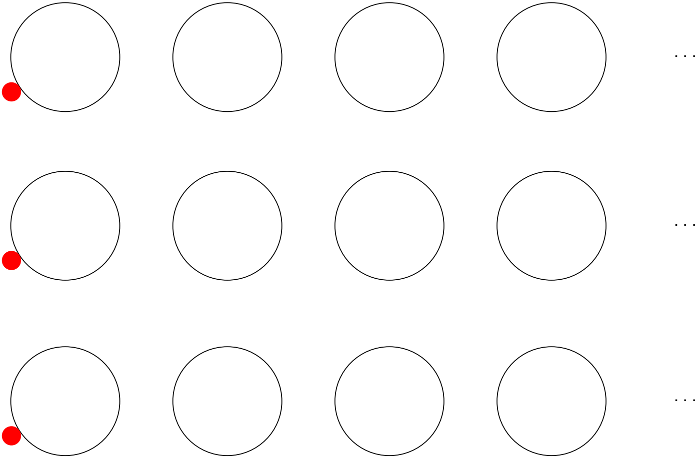
Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

Restaurant metaphor (1b)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



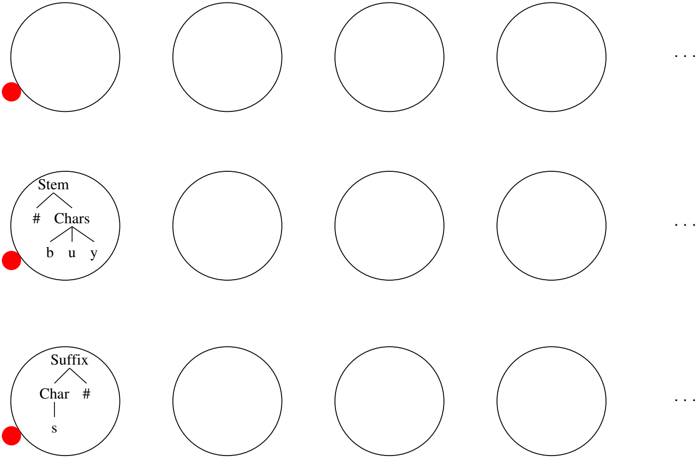
Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

Restaurant metaphor (1c)

Word restaurant
Word → Stem Suffix

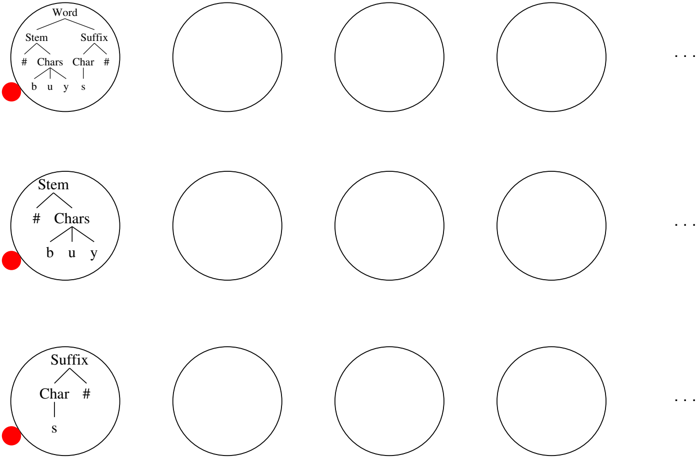
Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

Restaurant metaphor (1d)



Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #

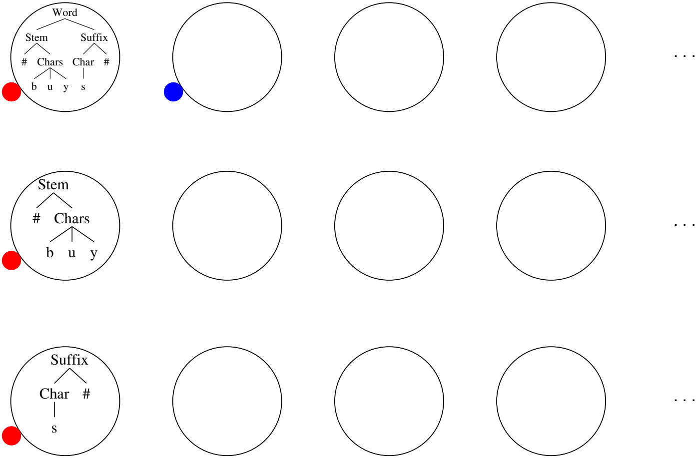
Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

Restaurant metaphor (2a)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



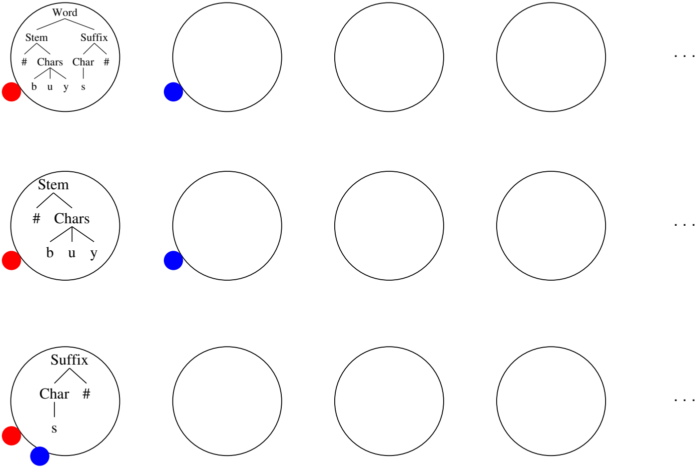
Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

Restaurant metaphor (2b)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



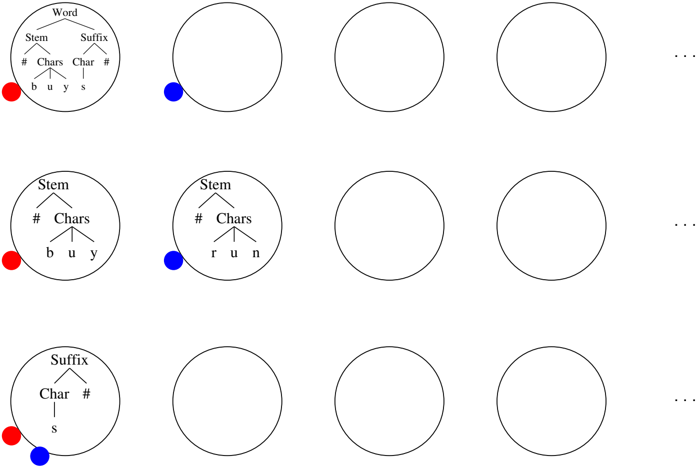
Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

Restaurant metaphor (2c)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

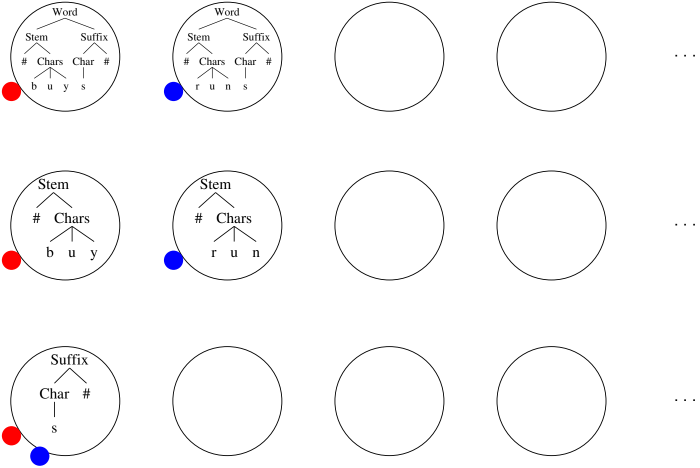
Restaurant metaphor (2d)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #

Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z



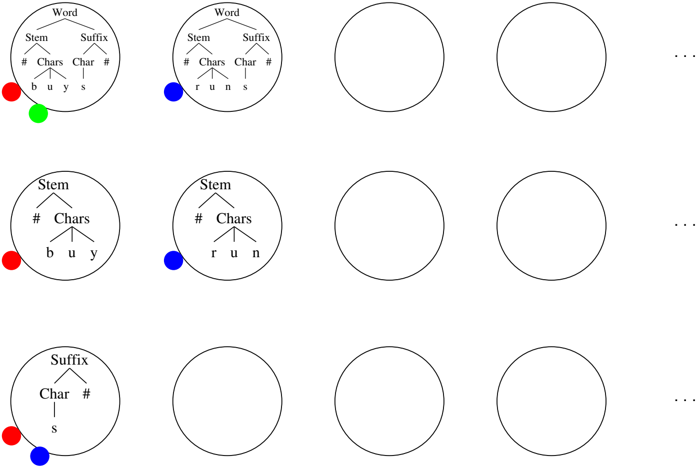
Restaurant metaphor (3)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #

Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z



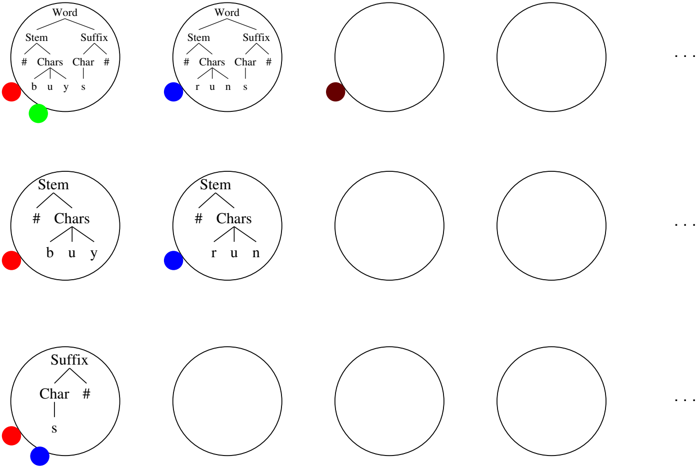
Restaurant metaphor (4a)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #

Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

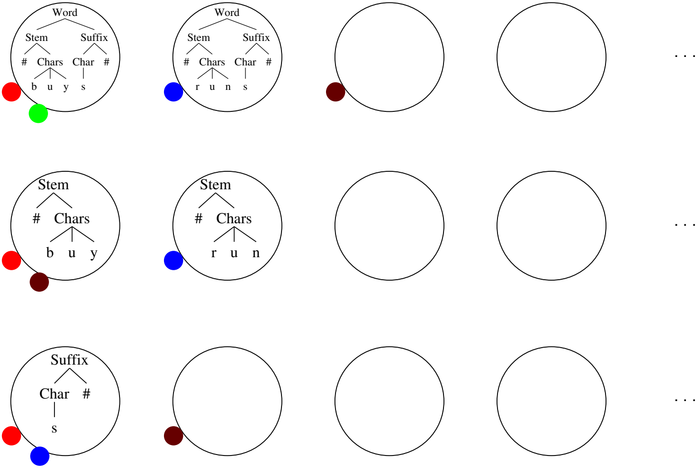


Restaurant metaphor (4b)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #



Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z

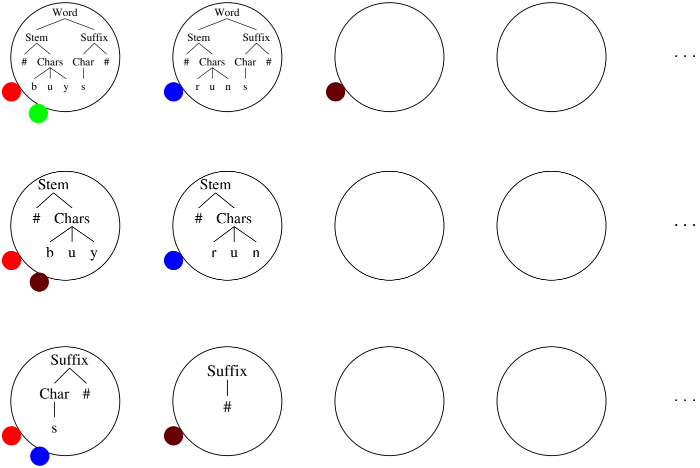
Restaurant metaphor (4c)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #

Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z



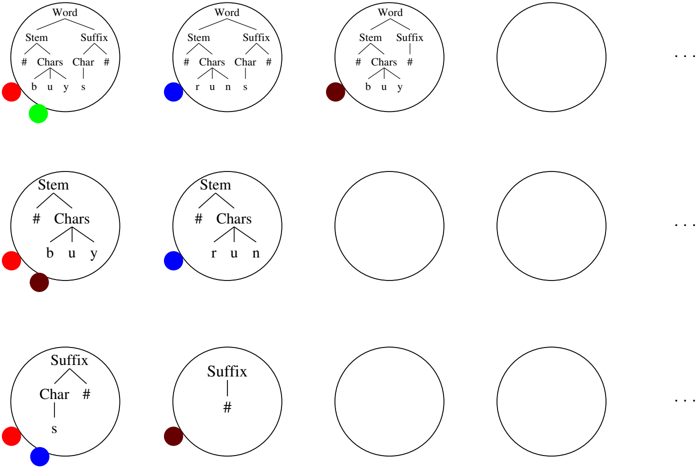
Restaurant metaphor (4d)

Word restaurant
Word → Stem Suffix

Stem restaurant
Stem → #
Stem → # Chars

Suffix restaurant
Suffix → #
Suffix → Chars #

Chars factory
Chars → Char
Chars → Char Chars
Char → a . . . z



Restaurants and Pitman-Yor processes

- Each restaurant is a Pitman-Yor process G_A (one per nonterminal $A \in N$)
- The customers' dinner are *samples* $y_{A,i} \sim G_A$
- The tables correspond to *indices* $k_{A,i}$, where $k_{A,i}$ is the table that customer i sits at
- The dinner on table k is $z_{A,k} \sim H_A$, where H_A is the label process for A
- $m_{A,i}$ is the number of tables occupied when restaurant A has i customers

Outline

Pitman-Yor processes

Each nonterminal $A \in N$ has a Pitman Yor process generating a sequence $y_{A,i} \sim G_A, i = 1, 2, \ldots$ of *labels* (trees with root labeled A). It does this by generating:

- a sequence $z_{A,k} \sim H_A, k = 1, 2, \ldots$ of labels drawn from base distribution H_A, where

$$H_A(z_{A,k}) = \sum_{z_{A,k}=uv} \sum_{A \to B\ C \in R_A} \theta_{A \to B\ C} G_B(u) G_C(v) + \sum_{A \to w \in R_A} \theta_{A \to w} \delta_w(z_{A,k})$$

- a sequence of *indices* $k_{A,i}$ into $z_{A,k}$ (positive integers)
- and setting $y_{A,i} = z_{A,k_{A,i}}$

Generating the indices k_i

- PY process generates samples $y_i = z_{k_i}$ from *base distribution samples* z_k and *indices* k_i
- Suppose we have already generated $k_1, \ldots, k_n$. Let:

$$\begin{aligned} m_n &= |\{k_i : i = 1, \ldots, n\}| && \text{(number of tables)} \\ n_k &= |\{i : k_i = k, i = 1, \ldots, n\}| && \text{(number of times } k_i = k\text{)} \end{aligned}$$

Then:

$$\begin{aligned} \mathrm{P}(k_{n+1} = k) &= \frac{n_k - a}{n + b} \quad \text{for } k \leq m_n && (k_{n+1} \text{ is old table}) \\ \mathrm{P}(k_{n+1} = m_n + 1) &= \frac{m_n a + b}{n + b} && (k_{n+1} \text{ is new table}) \end{aligned}$$

$\mathrm{P}(k_{n+1} | k_1, \ldots, k_n, z_1, \ldots, z_{m_n}, y_{n+1})$

- For MCMC, we incrementally generate tables k_i conditioned on observations y_i
- Given history $k_1, \ldots, k_n, z_1, \ldots, z_{m_n}$ and new label y_{n+1}, the probability of generating y_{n+1} via old table $k_{n+1} \leq m_n$ or new table $k_{n+1} = m_n + 1$ is:

$$
\begin{aligned}
& \mathrm{P}(k_{n+1} = k | k_1, \ldots, k_n, z_1, \ldots, z_{m_n}, y_{n+1}) \\
& \quad = \frac{n_k - a}{n + b} \, \delta_{z_k}(y_{n+1}) \qquad \text{(old tables)} \\
& \qquad + \frac{m_n a + b}{n + b} \, \delta_{m_n+1}(k) \, H(y_{n+1}) \qquad \text{(new table)}
\end{aligned}
$$

$$
\begin{aligned}
H_A(y) &= \sum_{y=uv} \sum_{A \to B\, C \in R_A} \theta_{A \to B\, C} G_B(u) G_C(v) \\
&\quad + \sum_{A \to w \in R_A} \theta_{A \to w} \delta_w(y)
\end{aligned}
$$

$G_A(y_{n+1}|k_1,\ldots,k_n,z_1,\ldots,z_{m_n})$

- MCMC sampling algorithms incrementally generate $k_1, k_2, \ldots$ and $y_1 = z_{k_1}, y_2 = z_{k_2}$

$$
\begin{aligned}
G_A(y_{n+1}|k_{1,n},z_{1,m_n}) &= \sum_{k=1}^{m_n} \frac{n_k - a_A}{n + b_A}\,\delta_{z_k}(y_{n+1}) && \text{(old tables)} \\
&\quad + \frac{m_n a_A + b_A}{n + b_A} H_A(y_{n+1}) && \text{(new table)} \\
H_A(y) &= \sum_{y=uv} \sum_{A\to B\,C \in R_A} \theta_{A\to B\,C}\, G_B(u) G_C(v) \\
&\quad + \sum_{A\to w \in R_A} \theta_{A\to w}\,\delta_w(y)
\end{aligned}
$$

Computation with grammars based on PY processes

- Given a history $k_{A,1}, \ldots, k_{A,n_A}, z_{A,1}, \ldots, z_{A,m_{A,n}}$, define a PCFG approximation (G', θ'_A) to G_A

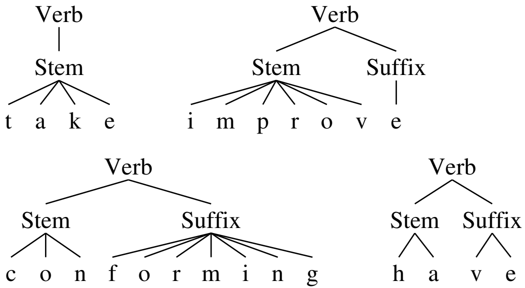
$$\begin{aligned}
\theta'_{A \to z_k} &= \frac{n_{A,k} - a_A}{n_A + b_A} \\
\theta'_{A \to B\,C} &= \frac{m_{n_A} a_A + b_A}{n_A + b_A}\, \theta_{A \to B\,C} \\
\theta'_{A \to w} &= \frac{m_{n_A} a_A + b_A}{n_A + b_A}\, \theta_{A \to w}
\end{aligned}$$

  - number of productions in $G' \propto$ number of tables m_{A,n_A}
  - history can change *within* a single tree, so in general (G', θ') is only an approximation
- sample a tree from PCFG (G', θ')
- Use Hastings acceptance/rejection to correct this to G_S

Outline

Verbal morphology

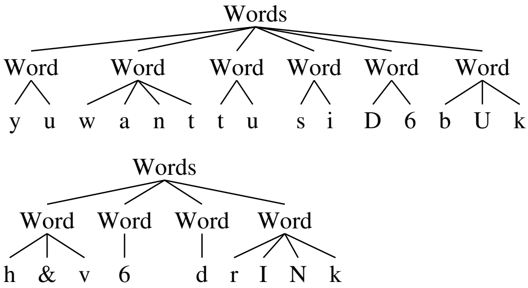
Verb → Stem
Verb → Stem Suffix
Stem → Chars
Suffix → Chars
Chars → Char
Chars → Char Chars
Char → a . . . z



- Input are orthographic verb tokens from WSJ
- Only cache (run restaurants for) Verb, Stem and Suffix; nodes with other labels not printed

Unigram model of word segmentation

Words → Word
Words → Word Words
Word → Chars
Chars → Char
Chars → Char Chars
Char → a . . . z



- Input is unsegmented broad phonemic transcription (Brent corpus)
- Only cache Word; nodes with other labels not printed

Morphology and word segmentation combined

$$
\begin{aligned}
&\text{Words} \rightarrow \text{Word} \\
&\text{Words} \rightarrow \text{Word Words} \\
&\text{Word} \rightarrow \text{Stem Suffix} \\
&\text{Word} \rightarrow \text{Stem} \\
&\text{Stem} \rightarrow \text{Chars} \\
&\text{Suffix} \rightarrow \text{Chars} \\
&\text{Chars} \rightarrow \text{Char} \\
&\text{Chars} \rightarrow \text{Char Chars} \\
&\text{Char} \rightarrow \text{a} \ldots \text{z}
\end{aligned}
$$

- Input is unsegmented broad phonemic transcription (Brent corpus)
- Only cache Word, Stem and Suffix; nodes with other labels not printed

Outline

Conclusion

- Overdispersion $\Rightarrow$ PCFGs are poor models of linguistic structure
- Estimating from *types* instead of *tokens* reduces overdispersion
  ...but is only possible in simple cases
- Pitman-Yor processes provide systematic way of downsampling tokens to types (or something in between)
- Define probability distribution over CFG trees by *associating each nonterminal with its own Pitman-Yor process*
  - CFG defines *possible structures*
  - Pitman-Yor process defines *probability of each (sub)structure*
- MCMC algorithms sample posterior tree distribution given strings
- Grammars based on PY processes recover linguistic structure where ML estimation of PCFGs fail