Bilexical Grammars
and a
$O(n^3)$ Probabilistic Parser

Jason Eisner
University of Pennsylvania
IWPT - 1997

Soft Selection

doff a **cap**
hat
sombrero
shirt
sink
clothe
about
...

Adjuncts too:
doffed his cap **to** her
at her
for her

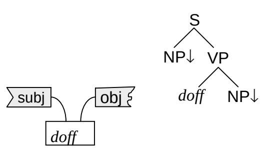
Lexicalized Grammars

doff: ___ NP

doff: (S\NP)/NP

S → NP *doff* NP

doff: [s, L(np), R(np)]



Rules are specialized for individual words
(or are implicit in lexical entries)

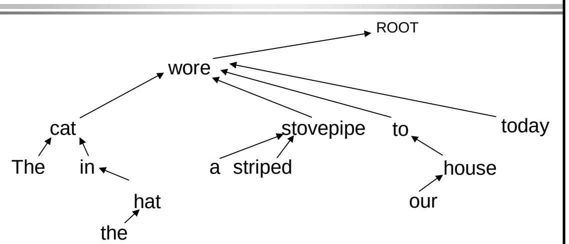
From **lexical** to **bilexical**

- Lafferty et al. 92, Charniak 95, Alshawi 96, Collins 96, Eisner 96, Goodman 97
- Also see Magerman 94, Ratnaparkhi 97, etc.

- Rules mention <u>two</u> words
  E.g., each verb can have its *own* distribution of arguments

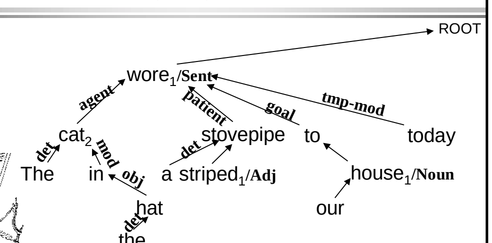
- Goal: No parsing performance penalty
  Alas, with standard chart parser:
  nonlexical $O(n^3)$
  lexical $O(n^5)$ ← *other methods give $O(n^4)$ or $O(n^3)$*
  bilexical $O(n^5)$

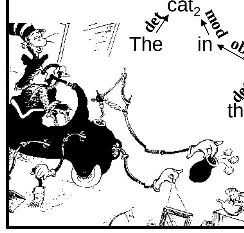
Simplified Formalism (1)



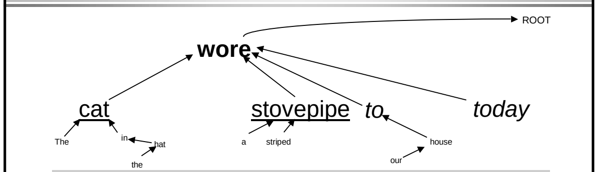
The cat in the hat wore a striped stovepipe to our house today.

(save these gewgaws for later)





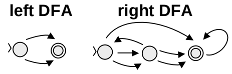
Simplified Formalism (2)



Need a flexible mechanism to score the possible sequences of dependents.

every lexical entry lists 2 idiosyncratic DFAs. These accept dependent sequences the word likes.

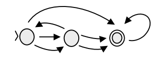
wore: **left DFA** **right DFA**



cat stovepipe to today

Weighting the Grammar

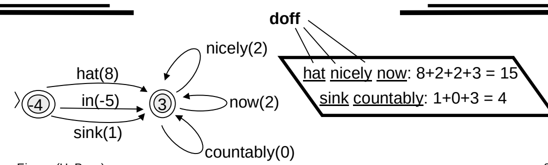
doff: **right DFA**



Transitive verb.
accepts: Noun Adv*

likes: hat nicely now (e.g., "[Bentley] **doffed** [his hat] [nicely] [just now]")

hates: sink countably (e.g., #"Bentley **doffed** [the sink] [countably]")



Why CKY is slow

1. visiting relatives is boring
2. visiting relatives wear funny hats
3. visiting relatives, we got bored and stole their funny hats

visiting relatives: $\underbrace{\text{NP(visiting), NP(relatives)}}$, AdvP, ...

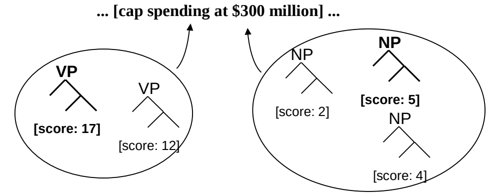
CFG says that all NPs are interchangeable
So we only have to use generic or best NP.

But bilexical grammar disagrees:
e.g., NP(visiting) is a poor subject for wear
We must try combining each analysis w/ context

Generic Chart Parsing (1)

- interchangeable analyses have same **signature**
- "analysis" = tree or dotted tree or ...

... [cap spending at $300 million] ...



- if ≤ S signatures, keep ≤ S analyses per substring

Generic Chart Parsing (2)

for each of the O($\underline{\mathbf{n}}^2$) substrings,
for each of O($\underline{\mathbf{n}}$) ways of splitting it,
for each of ≤ $\underline{\mathbf{S}}$ analyses of first half
for each of ≤ $\underline{\mathbf{S}}$ analyses of second half,
for each of ≤ $\underline{\mathbf{c}}$ ways of combining them:
combine, & add result to chart if best

$\boxed{O(n^3S^2c)}$

[cap spending] + [at $300 million] = [[cap spending] [at $300 million]]

≤ S analyses *≤ S analyses* *≤ cS^2 analyses of which we keep ≤ S*

Headed constituents ...

... have too many signatures.

How bad is $\Theta(n^3S^2c)$?

For unheaded constituents, S is constant: NP, VP ...
(similarly for dotted trees). So $\Theta(n^3)$.

But when different heads ⇒ different signatures,
the average substring has $\Theta(n)$ possible heads
and S=$\Theta(n)$ possible signatures. So $\Theta(n^5)$.

Forget heads - think hats!

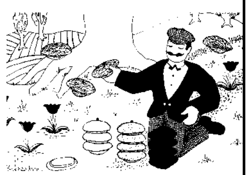
Solution:

Don’t assemble the parse from <u>constituents</u>.

Assemble it from <u>**spans**</u> instead.

wore → ROOT
cat → wore
The → cat
in → cat
hat → in
the → hat
stovepipe → wore
a → stovepipe

The cat in the hat wore a stovepipe. ROOT



Spans vs. constituents

Two kinds of substring.

» <u>**Constituent**</u> of the tree: links to the rest only through its <u>head</u>.

The <u>cat</u> in the hat wore a stovepipe. ROOT

» <u>**Span**</u> of the tree: links to the rest only through its <u>endwords</u>.

The <u>cat</u> in the hat <u>wore</u> a stovepipe. ROOT

Decomposing a tree into spans

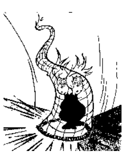
The cat in the hat wore a stovepipe. ROOT

The cat + cat in the hat wore a stovepipe. ROOT

cat in the hat wore + wore a stovepipe. ROOT

cat in + in the hat wore

in the hat + hat wore



Maintaining weights

Seed chart w/ word <u>pairs</u> x y , x y , x y

Step of the algorithm:

a ... b + b ... c = { a ... b ... c ; a ... b ... c ; a ... b ... c }

We can add an arc only if a, c are both parentless.

weight(a ... b ... c) = weight(a ... b) + weight(b ... c)

+ weight of c arc from a’s right DFA state

+ weights of stopping in b’s left and right states

Analysis

Algorithm is $O(n^3 S^2)$ time, $O(n^2 S)$ space. What is S?

a ... b + b ... c = a ... b ... c

Where:
- »b gets a parent from exactly one side
- »Neither a nor c previously had a parent
- »a’s right DFA accepts c; b’s DFAs can halt

Signature of a ... b has to specify parental status & DFA state of a and b

∴ $S = O(t^2)$ where t is the maximum # states of any DFA

S independent of n because all of a substring’s analyses are headed in the same place - at the ends!

Improvement

Can reduce S from $O(t^2)$ to $O(t)$

a ... b + b ... c = a ... b ... c

state of b*’s left automaton tells us weight of halting*

likewise for b*’s right automaton*

The halt-weight for each half is *independent* of the other half.

Add every span to both <u>left chart</u> & <u>right chart</u>

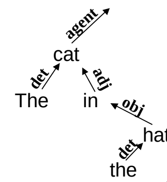
Above, we draw a ... b from left chart, b ... c from right chart

Copy of a ... b in left chart has halt weight for b already added so its signature needn’t mention the state of b’s automaton

Embellishments

- More detailed parses
  - » Labeled edges
  - » Tags (part of speech, word sense, ...)
  - » Nonterminals
- How to encode probability models

More detailed parses (1)



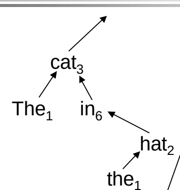
Labeled arcs

Grammar:
DFAs must accept strings of word-role pairs
e.g., (cat, **agent**) or (hat, **obj**)

Parser:
When we stick two spans together, consider covering with:
nothing, agent , agent , obj , obj
etc.

Time penalty:
O(m) where m is the number of label types

More detailed parses (2)



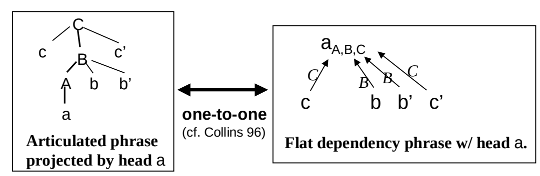
Optimize tagging & parsing at once

Grammar:
Every input token denotes confusion set
cat = {cat_1, cat_2, cat_3, cat_4}
Choice of cat_3 adds a certain weight to parse

Parser:
More possibilities for seeding chart
Tags of b must match in $\boxed{a \dots b}$ + $\boxed{b \dots c}$
Signature of $\boxed{a \dots b}$ must specify tags of a and b

Time penalty:
$O(g^4)$ where g is max # tags per input word
since S goes up by g^2
$O(g^3)$ by considering only appropriate $\boxed{b \dots c}$

Nonterminals



Articulated phrase projected by head a

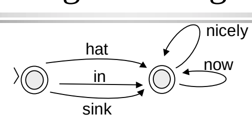
Flat dependency phrase w/ head a.

The bilexical DFAs for $a_{A,B,C}$ insist that its kids come in order *A, B, C.*

Use fast bilexical algorithm, then convert result to nonterminal tree.
Want small (and finite) set of tags like $a_{A,B,C}$.
(Guaranteed by X-bar theory: doff = {$doff_{V,\bar{V},VP}$, $doff_{V,\bar{V},VP,S}$}.)

Using the weights

doff:

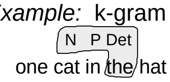


- *Deterministic grammar:* All weights 0 or -∞
- *Generative model:*
  log Pr(next kid = *nicely* | *doff* in state 2)
- *Comprehension model:*
  log Pr(next kid = *nicely* | *doff* in state 2, *nicely* present)
- **Eisner 1996** compared several models, found significant differences

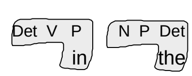
String-local constraints

Seed chart with word pairs like $\boxed{x \ \ y}$
We can choose to exclude some such pairs.

Example: k-gram tagging. (here k=3)



tag with part-of-speech *trigrams*
weight = log Pr(the | Det)Pr(Det | N,P)



excluded bigram:
the 2 words disagree on tag for “cat”

Conclusions

- Bilexical grammar formalism
  - How much do 2 words want to relate?
  - Flexible: encode your favorite representation
  - Flexible: encode your favorite prob. model
- Fast parsing algorithm
  - Assemble spans, not constituents
  - $O(n^3)$, not $O(n^5)$. Precisely, $O(n^3t^2g^3m)$.
    - t=max DFA size, g=max senses/word, m=# label types
    - These grammar factors are typically small