Efficient Parsing for
- Bilexical CF Grammars
- Head Automaton Grammars

Jason Eisner
U. of Pennsylvania
U. of Rochester

Giorgio Satta
U. of Padova, Italy

When's a grammar bilexical?

If it has rules / entries that mention 2 specific words in a dependency relation:

convene - meeting
eat - blintzes
ball - bounces
joust - with

Bilexical Grammars

- Instead of **VP → V NP**
- or even **VP → solved NP**
- use detailed rules that mention **2 heads**:
  - **S[solved] → NP[Peggy] VP[solved]**
  - **VP[solved] → V[solved] NP[puzzle]**
  - **NP[puzzle] → Det[a] N[puzzle]**
- so we can exclude, or reduce probability of,
  - **VP[solved] → V[solved] NP[goat]**
  - **NP[puzzle] → Det[two] N[puzzle]**

Bilexical CF grammars

- Every rule has one of these forms:
  - **A[x] → B[x] C[y]** *so head of LHS*
  - **A[x] → B[y] C[x]** *is inherited from*
  - **A[x] → x** *a child on RHS.*
- (rules could also have probabilities)

B[x], B[y], C[x], C[y], ... many nonterminals
A, B, C ... are "traditional nonterminals"
x, y ... are words

Bilexicalism at Work

- Not just selectional but adjunct preferences:
  - Peggy [solved a puzzle] from the library.
  - Peggy solved [a puzzle from the library].

Hindle & Rooth (1993) - PP attachment

Bilexicalism at Work

Bilexical parsers that fit the CF formalism:

Alshawi (1996) - head automata
Charniak (1997) - Treebank grammars
Collins (1997) - context-free grammars
Eisner (1996) - dependency grammars

Other superlexicalized parsers that don't:

Jones & Eisner (1992) - bilexical LFG parser
Lafferty et al. (1992) - stochastic link parsing
Magerman (1995) - decision-tree parsing
Ratnaparkhi (1997) - maximum entropy parsing
Chelba & Jelinek (1998) - shift-reduce parsing

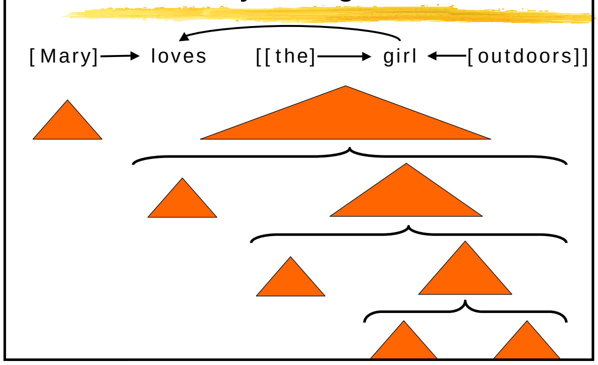
How bad is bilex CF parsing?

A[x] → B[x] C[y]

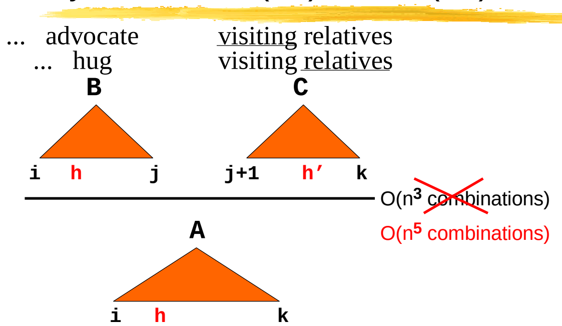
- Grammar size = $O(t^3 V^2)$
  where t = |{**A**, **B**, …}| V = |{**x**, **y** …}|
- So CKY takes $\mathbf{O(t^3 V^2 n^3)}$
- Reduce to $\mathbf{O(t^3 n^5)}$ since relevant V = n

- This is terrible … can we do better?
  - Recall: regular CKY is $O(t^3 n^3)$

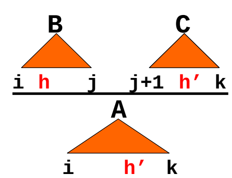
The CKY-style algorithm



Why CKY is $O(n^5)$ not $O(n^3)$



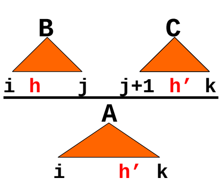
Idea #1

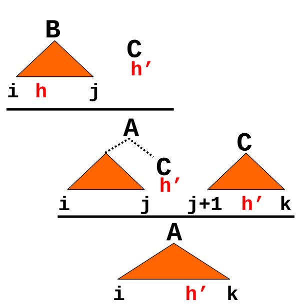


- Combine B with what C?
  - must try different-width C's (vary **k**)
  - must try differently-headed C's (vary **h′**)
  - Separate these!

Idea #1

(the old CKY way)





Head Automaton Grammars

(Alshawi 1996)

[Good old Peggy] **solved** [the puzzle] [with her teeth] !

The **head automaton** for **solved**:
- a finite-state device
- can consume words adjacent to it on either side
- does so after they've consumed *their* dependents

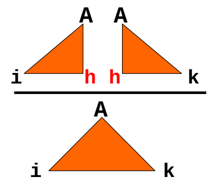
[Peggy] **solved** [puzzle] [with]	(state = V)
[Peggy] **solved** [with]	(state = VP)
[Peggy] **solved**	(state = VP)
solved	(state = S; halt)

Formalisms too powerful?

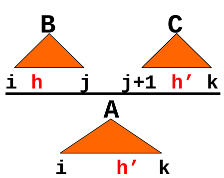
- So we have Bilex CFG and HAG in $O(n^4)$.
- HAG is quite powerful - head c can require a^n **c** b^n:
  ... $[...a_3...]$ $[...a_2...]$ $[...a_1...]$ **c** $[...b_1...]$ $[...b_2...]$ $[...b_3...]$ ...
  not center-embedding, `[a₃ [[a₂ [[a₁] b₁]] b₂]] b₃`
  - Linguistically unattested and unlikely
  - Possible only if the HA has a left-right cycle
  - **Absent such cycles, can we parse faster?**
    - (for both HAG and equivalent Bilexical CFG)

Transform the grammar

- Absent such cycles,
  we can transform to a "split grammar":
  - Each head eats all its right dependents first
  - I.e., left dependents are more oblique.
- This allows



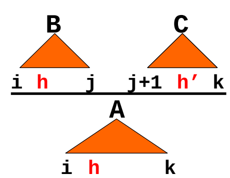
Idea #2

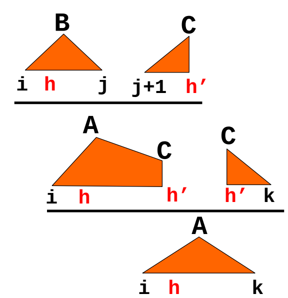


- Combine what B and C?
  - must try different-width C's (vary **k**)
  - must try different midpoints **j**
  - Separate these!

Idea #2

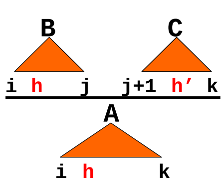
(the old CKY way)

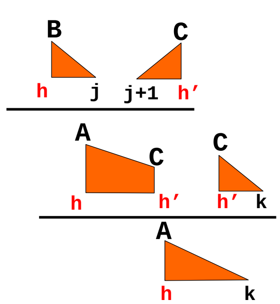




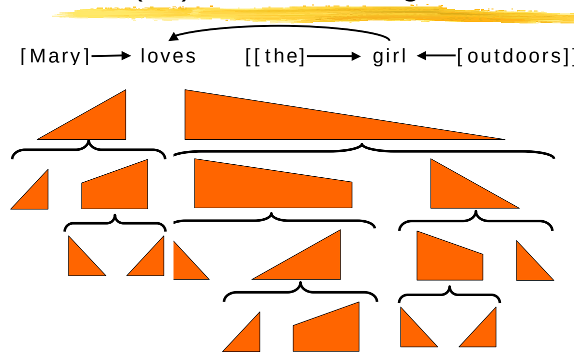
Idea #2

(the old CKY way)





The $O(n^3)$ half-tree algorithm



Theoretical Speedup

- **n** = input length **g** = polysemy
- **t** = traditional nonterms or automaton states
- Naive: $\mathbf{O(n^5 g^2 t)}$
- New: $\mathbf{O(n^4 g^2 t)}$
- Even better for split grammars:
  - Eisner (1997): $\mathbf{O(n^3 g^3 t^2)}$
  - New: $\mathbf{O(n^3 g^2 t)}$

  all independent of vocabulary size!

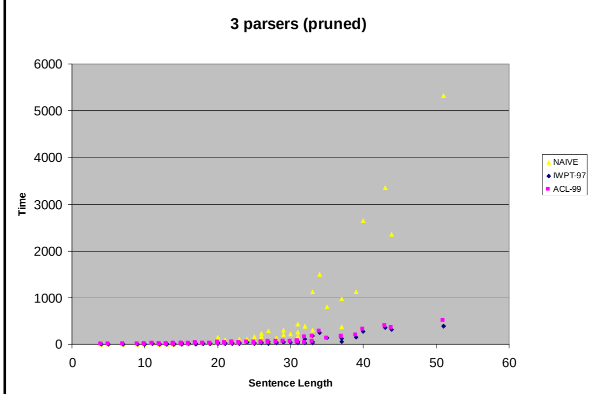
Reality check

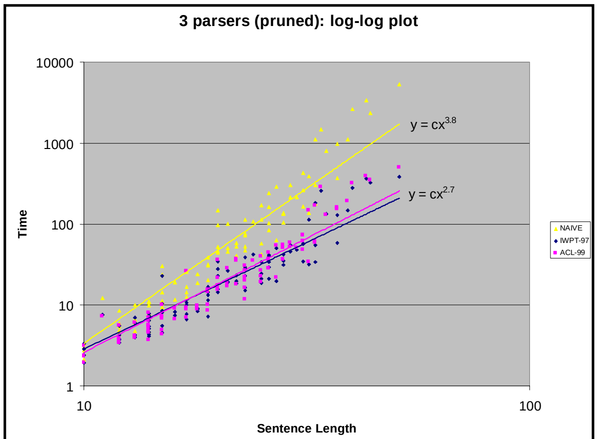
- Constant factor
- Pruning may do just as well
  - "visiting relatives": 2 plausible NP hypotheses
  - i.e., both heads survive to compete - common??
- Amdahl's law
  - much of time spent smoothing probabilities
  - fixed cost per parse if we cache probs for reuse

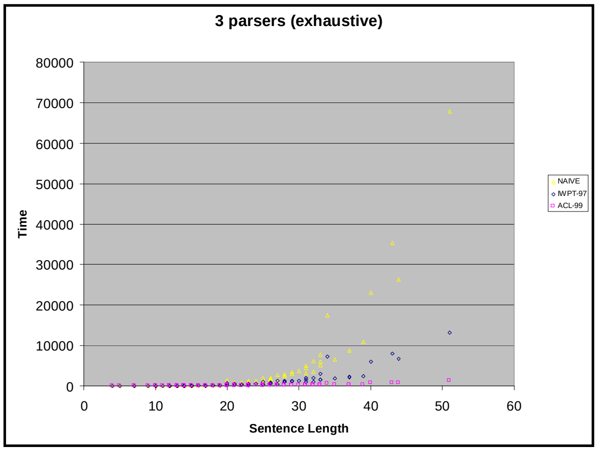
Experimental Speedup (not in paper)

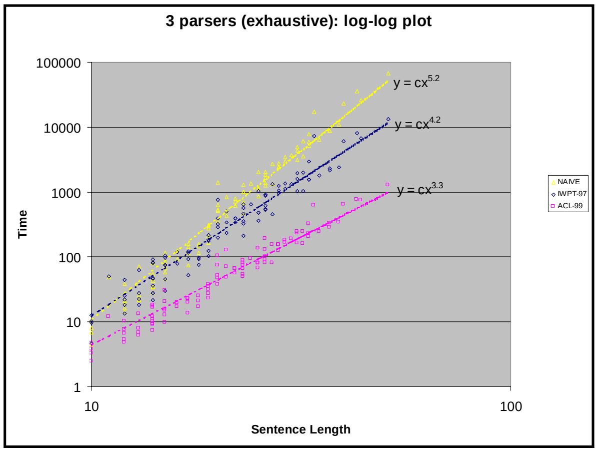
Used Eisner (1996) Treebank WSJ parser
and its split bilexical grammar

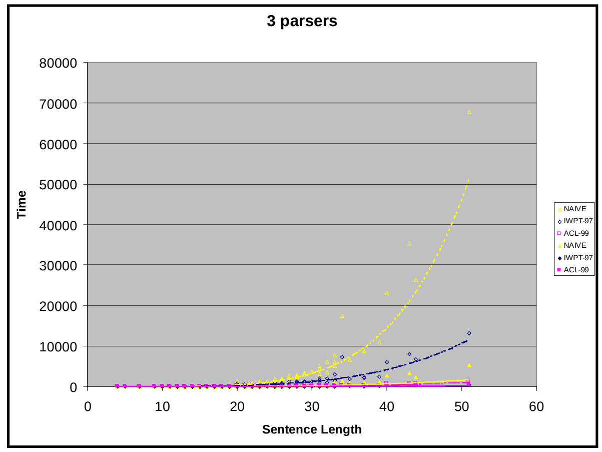
- Parsing with pruning:
  - Both old and new $O(n^3)$ methods give **5x** speedup over the $O(n^5)$ - at 30 words
- Exhaustive parsing (e.g., for EM):
  - Old $O(n^3)$ method (Eisner 1997) gave **3x** speedup over $O(n^5)$ - at 30 words
  - **New $O(n^3)$ method gives 19x speedup**

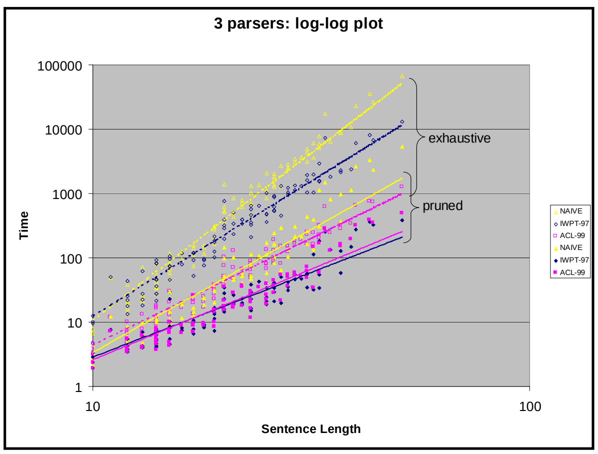












Summary

- Simple bilexical CFG notion **A[x] → B[x] C[y]**
- Covers several existing stat NLP parsers

- Fully general $O(n^4)$ algorithm - not $O(n^5)$
- Faster $O(n^3)$ algorithm for the “split” case
- Demonstrated practical speedup

- *Extensions:* TAGs and post-transductions