Efficient Parsing for
• Bilexical CF Grammars
• Head Automaton Grammars

Jason Eisner
U. of Pennsylvania

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 \( VP \rightarrow V \ NP \)
• or even \( VP \rightarrow \text{solved} \ NP \)
• use detailed rules that mention 2 heads:

\[
\begin{align*}
S(\text{solved}) & \rightarrow \text{NP} [\text{Peggy}] \ VP(\text{solved}) \\
VP(\text{solved}) & \rightarrow V(\text{solved}) \ NP [\text{puzzle}] \\
NP [\text{puzzle}] & \rightarrow \text{Det} [a] \ N [\text{puzzle}]
\end{align*}
\]

- so we can exclude, or reduce probability of,

\[
\begin{align*}
VP(\text{solved}) & \rightarrow V(\text{solved}) \ NP [\text{goat}] \\
NP [\text{puzzle}] & \rightarrow \text{Det} [\text{two}] \ N [\text{puzzle}]
\end{align*}
\]

Bilexical CF grammars

• Every rule has one of these forms:

\[
A(x) \rightarrow B[x] \ C[y] \quad \text{so head of LHS}
\]

\[
A(x) \rightarrow B[y] \ C[x] \quad \text{is inherited from}
\]

a child on RHS.

(rules could also have probabilities)

\[
B[x], B[y], C[x], C[y], \ldots \text{ many nonterminals}
\]

\[
A, B, C \ldots \text{ are “traditional nonterminals”}
\]

\[
x, y \ldots \text{ are words}
\]

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?

- Grammar size = $O(t^3 V^2)$
  
  where $t = |\{A, B, \ldots\}|$  
  $V = |\{x, y \ldots\}|$

- So CKY takes $O(t^3 V^2 n^3)$
- Reduce to $O(t^3 n^5)$ since relevant $V = n$

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

The CKY-style algorithm

\[
\text{Mary} \rightarrow \text{loves} \left[ \left[ \text{the} \right] \rightarrow \left[ \text{girl} \leftarrow \left[ \text{outdoors} \right] \right] \right]
\]

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

\[
... \ \text{advocate} \ \text{visiting relatives} \\
... \ \text{hug} \ \text{visiting relatives}
\]

\[
A[i \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow B[i] \ C[y]
\]

\[
\text{Grammar size} = O(t^3 V^2)
\]

\[
\text{So CKY takes} \ O(t^3 V^2 n^3)
\]

\[
\text{Reduce to} \ O(t^3 n^5) \ \text{since relevant} \ V = n
\]

\[
\text{This is terrible} \ ... \ \text{can we do better?}
\]

\[
\text{Recall: regular CKY is} \ O(t^3 n^3)
\]

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

  * Separate these!

Head Automaton Grammars

(Alshawi 1996)

[Good old Peggy \textbf{solved} [the puzzle] [with her teeth] !

The \textbf{head automaton} for \textbf{solved}:

- a finite-state device
- can consume words adjacent to it on either side
- does so after they've consumed their dependents

\[
\begin{align*}
\text{(Peggy) } & \text{\textbf{solved}} \ [\text{puzzle}] \ [\text{with}] \quad \text{(state = V)} \\
\text{(Peggy) } & \text{\textbf{solved}} \ [\text{with}] \quad \text{(state = VP)} \\
\text{(Peggy) } & \text{\textbf{solved}} \ \text{\textbf{solved}} \quad \text{(state = S; halt)}
\end{align*}
\]
Formalisms too powerful?

- So we have Bilex CFG and HAG in $O(n^4)$.
- HAG is quite powerful - head $c$ can require $a^p b^q$:
  $\ldots [\ldots a_3 \ldots] [\ldots a_2 \ldots] [\ldots a_1 \ldots] [\ldots b_1 \ldots] [\ldots b_2 \ldots] [\ldots b_3 \ldots] \ldots$
  not center-embedding. $[a_3 [a_2 [a_1 b_1] b_2] b_3]$
- 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

[Mary]→ loves [[[the]] girl →[outdoors]]
### Theoretical Speedup

- \( n \) = input length
- \( g \) = polysemy
- \( t \) = traditional nonterms or automaton states

- Naive: \( O(n^5 g^2 t) \)
- **New**: \( O(n^4 g^2 t) \)

Even better for split grammars:
- Eisner (1997): \( O(n^3 g^3 t^2) \)
- **New**: \( O(n^3 g^2 t) \)

*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

---

**3 parsers (pruned): log-log plot**

- \( y = cx^{3.8} \)
- \( y = cx^{2.7} \)

---

**3 parsers (exhaustive)**

- \( y = cx^{2.7} \)
Summary

- Simple bilexical CFG notion $A \to A [x] \to B [x] C [y]$
- 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