# Toward a Psycholinguistically-Motivated Model of Language Processing

William Schuler[1],
Samir AbdelRahman[2],
Tim Miller[1],
Lane Schwartz[1]

June 24, 2011

[1]University of Minnesota
[2]Cairo University

# Background

NSF project: implement interactive model of speech/language processing

- Parsing/speech recognition dep. on semantic interpretation in context (Tanenhaus et al., 1995, 2002)

# Background

NSF project: implement interactive model of speech/language processing

- Parsing/speech recognition dep. on semantic interpretation in context (Tanenhaus et al., 1995, 2002)
- Factored time-series model of speech recognition, parsing, interpretation (formal model presented in Computational Linguistics, in press)
- Real-time interactive speech interface: define new objects, then refer (implemented system presented at IUI'08; interp. → vectors of objects)
- This year: interp. vector → head word probabilities / LSA semantics

# Background

NSF project: implement interactive model of speech/language processing

- Parsing/speech recognition dep. on semantic interpretation in context (Tanenhaus et al., 1995, 2002)
- Factored time-series model of speech recognition, parsing, interpretation (formal model presented in Computational Linguistics, in press)
- Real-time interactive speech interface: define new objects, then refer (implemented system presented at IUI'08; interp. → vectors of objects)
- This year: interp. vector → head word probabilities / LSA semantics
- Why time-series? composition expensive; time-series simpler than CKY

# Background

NSF project: implement interactive model of speech/language processing

- Parsing/speech recognition dep. on semantic interpretation in context (Tanenhaus et al., 1995, 2002)
- Factored time-series model of speech recognition, parsing, interpretation (formal model presented in Computational Linguistics, in press)
- Real-time interactive speech interface: define new objects, then refer (implemented system presented at IUI'08; interp. → vectors of objects)
- This year: interp. vector → head word probabilities / LSA semantics
- Why time-series? composition expensive; time-series simpler than CKY
- Today: is it safe? Human-like memory limits still parse most sentences (evaluated on broad-coverage WSJ Treebank)

# Background

NSF project: implement interactive model of speech/language processing

- Parsing/speech recognition dep. on semantic interpretation in context (Tanenhaus et al., 1995, 2002)
- Factored time-series model of speech recognition, parsing, interpretation (formal model presented in Computational Linguistics, in press)
- Real-time interactive speech interface: define new objects, then refer (implemented system presented at IUI'08; interp. → vectors of objects)
- This year: interp. vector → head word probabilities / LSA semantics
- Why time-series? composition expensive; time-series simpler than CKY
- Today: is it safe? Human-like memory limits still parse most sentences (evaluated on broad-coverage WSJ Treebank)
- Friday: model transform also gives nice explanation of speech repair (evaluated on Switchboard Treebank)

# Parsing in Short-term Memory

**Early work:**

Marcus ('80), Abney & Johnson ('91), Gibson ('91), Lewis ('93), ... —
Garden pathing, processing difficulties due to memory saturation

- processing difficulties also due to other factors,
  e.g. similarity (Miller & Chomsky '63; Lewis '93), decay (Gibson '98)
- favor left-corner; but eager/deferred composition? → parallel proc.

# Parsing in Short-term Memory

**Early work:**

Marcus ('80), Abney & Johnson ('91), Gibson ('91), Lewis ('93), ... —
Garden pathing, processing difficulties due to memory saturation

- processing difficulties also due to other factors,
  e.g. similarity (Miller & Chomsky '63; Lewis '93), decay (Gibson '98)
- favor left-corner; but eager/deferred composition? → parallel proc.

**More recently:**

Hale (2003), Levy (2008) —
Difficulties due to changing probability/activation of competing hypotheses

- empirical success
- decouples processing difficulty from memory saturation
- but does not explain how/whether parsing fits in short-term memory
  (and parsing should now be comfortably within STM, not at limit!)

# Parsing in Short-term Memory

**This model:**

Explicit memory elements, compatible w. interactive interpretation

- Bounded store of incomplete referents, constituents over time
  - incomplete referets: individual/group of objects/events ($\sim$ Haddock'89)
  - incomplete constituents: e.g. S/NP (S w/o NP; $\sim$ CCG, Steedman'01)

# Parsing in Short-term Memory

**This model:**

Explicit memory elements, compatible w. interactive interpretation

- Bounded store of incomplete referents, constituents over time
  - incomplete referets: individual/group of objects/events ($\sim$ Haddock'89)
  - incomplete constituents: e.g. S/NP (S w/o NP; $\sim$ CCG, Steedman'01)
- For simplicity, strict complexity limit on memory elements (no chunks):
  one incomplete referent/constituent per memory element

# Parsing in Short-term Memory

**This model:**

Explicit memory elements, compatible w. interactive interpretation

- Bounded store of incomplete referents, constituents over time
  - incomplete referets: individual/group of objects/events ($\sim$ Haddock'89)
  - incomplete constituents: e.g. S/NP (S w/o NP; $\sim$ CCG, Steedman'01)
- For simplicity, strict complexity limit on memory elements (no chunks):
  one incomplete referent/constituent per memory element
- Sequence of stores $\Leftrightarrow$ phrase structure via simple tree transform ($\sim$Johnson'98; system $\sim$Roark'01/Henderson'04 but mem-optimized)

# Parsing in Short-term Memory

**This model:**

Explicit memory elements, compatible w. interactive interpretation

- Bounded store of incomplete referents, constituents over time
  - incomplete referets: individual/group of objects/events (∼ Haddock'89)
  - incomplete constituents: e.g. S/NP (S w/o NP; ∼ CCG, Steedman'01)
- For simplicity, strict complexity limit on memory elements (no chunks):
  one incomplete referent/constituent per memory element
- Sequence of stores ⇔ phrase structure via simple tree transform (∼Johnson'98; system ∼Roark'01/Henderson'04 but mem-optimized)
- Alternative stores active in pockets, not monolithic (unbounded beam)
- Essentially, factored HMM-like time-series model

# Parsing in Short-term Memory

**This model:**

Explicit memory elements, compatible w. interactive interpretation

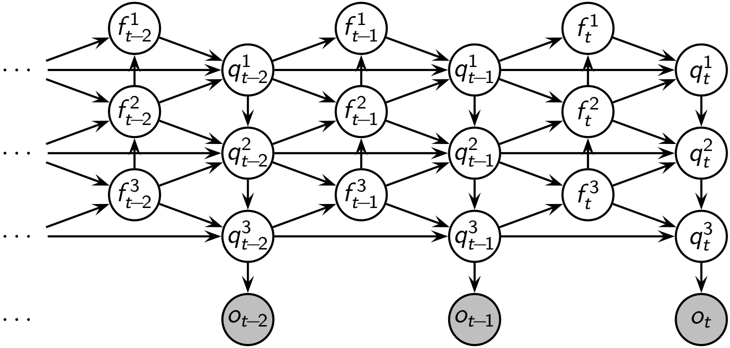
- Bounded store of incomplete referents, constituents over time
  - incomplete referets: individual/group of objects/events (∼ Haddock'89)
  - incomplete constituents: e.g. S/NP (S w/o NP; ∼ CCG, Steedman'01)
- For simplicity, strict complexity limit on memory elements (no chunks):
  one incomplete referent/constituent per memory element
- Sequence of stores ⇔ phrase structure via simple tree transform (∼Johnson'98; system ∼Roark'01/Henderson'04 but mem-optimized)
- Alternative stores active in pockets, not monolithic (unbounded beam)
- Essentially, factored HMM-like time-series model

**Evaluation of Coverage:**

- Can parse nearly 99.96% of WSJ 2–21 using $\leq$ 4 memory elements

# Hierarchic Hidden Markov Model

Factored HMM model (Murphy & Paskin '01): bounded probabilistic PDA

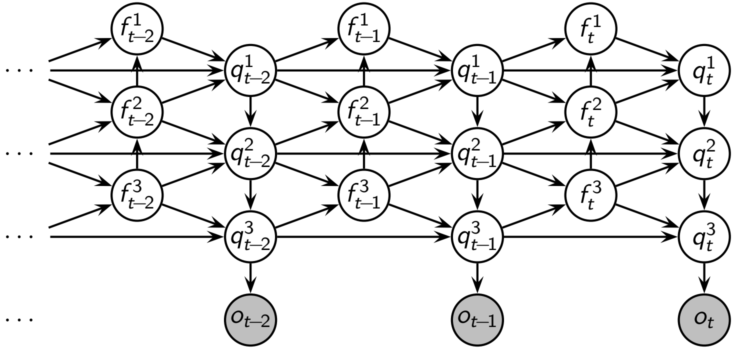


Hidden syntax+ref model, generating observations: words / acoust. features

$$\hat{h}^{1..D}_{1..T} \stackrel{\text{def}}{=} \underset{h^{1..D}_{1..T}}{\operatorname{argmax}} \prod_{t=1}^{T} \mathsf{P}_{\Theta_{\mathsf{LM}}}(h^{1..D}_{t} \mid h^{1..D}_{t-1}) \cdot \mathsf{P}_{\Theta_{\mathsf{OM}}}(o_t \mid h^{1..D}_{t})$$

# Hierarchic Hidden Markov Model

Factored HMM model (Murphy & Paskin '01): bounded probabilistic PDA

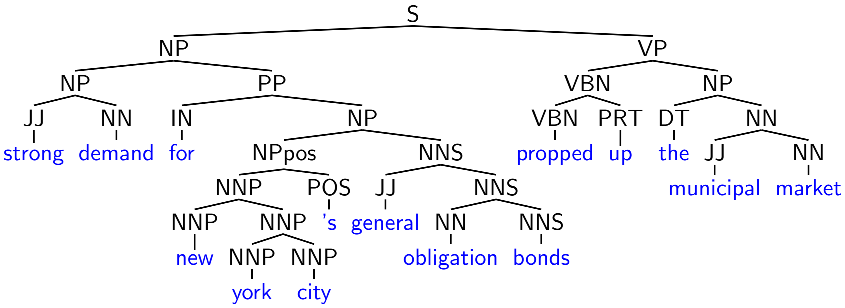



$$\mathsf{P}_{\Theta_{\mathrm{LM}}}(q_t^{1..D} \mid q_{t\text{-}1}^{1..D}) = \sum_{f_t^{1..D}} \mathsf{P}_{\Theta_{\mathrm{Reduce}}}(f_t^{1..D} \mid q_{t\text{-}1}^{1..D}) \cdot \mathsf{P}_{\Theta_{\mathrm{Shift}}}(q_t^{1..D} \mid f_t^{1..D}\ q_{t\text{-}1}^{1..D})$$

$$\stackrel{\mathrm{def}}{=} \sum_{f_t^{1..D}} \prod_{d=1}^{D} \mathsf{P}_{\Theta_\rho}(f_t^d \mid f_t^{d+1} q_{t\text{-}1}^d q_{t\text{-}1}^{d\text{-}1}) \cdot \mathsf{P}_{\Theta_\sigma}(q_t^d \mid f_t^{d+1} f_t^d\ q_{t\text{-}1}^d q_t^{d\text{-}1})$$

# Saving Memory with a Transformed Grammar

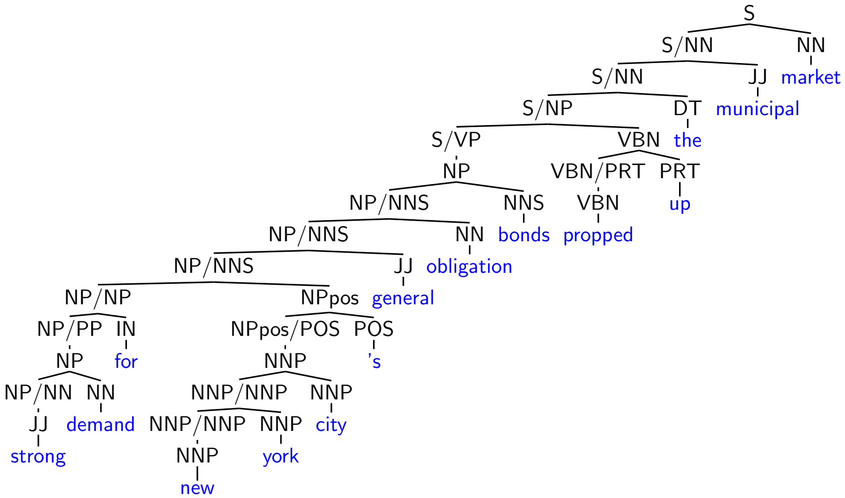
Derive model probabilities from training trees:




Must be transformed into flat, memory-efficient form...

# Saving Memory with a Transformed Grammar

'Right-corner transform': ∼ left-corner, but reversed so incomplete on right



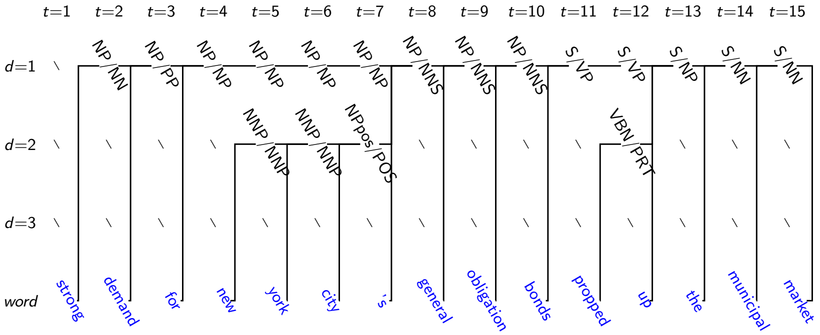
# Mapping to HHMM

Align levels to a grid, to train HHMM:

| | $t$=1 | $t$=2 | $t$=3 | $t$=4 | $t$=5 | $t$=6 | $t$=7 | $t$=8 | $t$=9 | $t$=10 | $t$=11 | $t$=12 | $t$=13 | $t$=14 | $t$=15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $d$=1 | \ | NP/NN | NP/PP | NP/NP | NP/NP | NP/NP | NP/NP | NP/NNS | NP/NNS | NP/NNS | S/VP | S/VP | S/NP | S/NN | S/NN |
| $d$=2 | \ | \ | \ | \ | NNP/NNP | NNP/NNP | NPpos/POS | \ | \ | \ | \ | VBN/PRT | \ | \ | \ |
| $d$=3 | \ | \ | \ | \ | \ | \ | \ | \ | \ | \ | \ | \ | \ | \ | \ |
| *word* | strong | demand | for | new | york | city | 's | general | obligation | bonds | propped | up | the | municipal | market |

# Mapping to HHMM

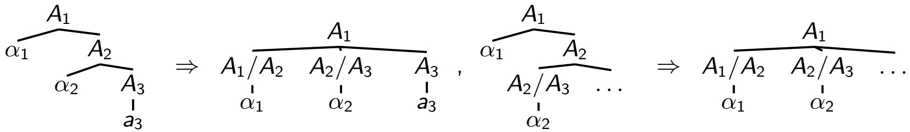
Align levels to a grid, to train HHMM:



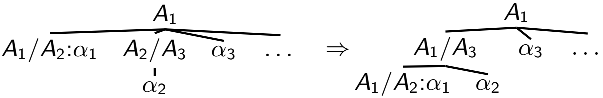

Different than other left-corner models: not all levels open for adjunction
Many configs in parallel; weights depend on learned HHMM probabilities.

# Tree Transform

Transform is very simple — first flatten out right-recursive structure:

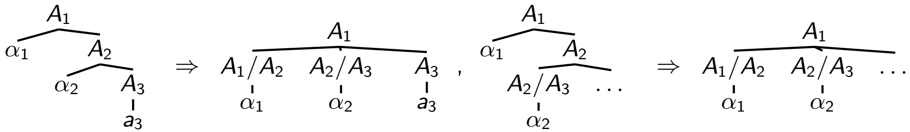



then replace it with left-recursive structure:

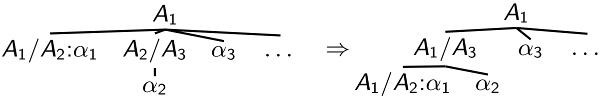


# Tree Transform

Transform is very simple — first flatten out right-recursive structure:




then replace it with left-recursive structure:




Only right recursion remaining is center embedding, known to be limited:
"The cart the horse the man bought pulled broke."
(Miller and Chomsky, 1963)

# Coverage

How many levels do you need? About four.

| stack memory capacity | sentences | coverage |
|---|---|---|
| no stack memory | 127 | 0.32% |
| 1 stack element | 3,496 | 8.78% |
| 2 stack elements | 25,909 | 65.05% |
| 3 stack elements | 38,902 | 97.67% |
| **4 stack elements** | **39,816** | **99.96%** |
| 5 stack elements | 39,832 | 100.00% |
| TOTAL | 39,832 | 100.00% |

Percent coverage of transformed treebank sections 2–21 w. no punctuation

Good! Because that's supposed to be our limit! (Cowan, 2001)

# Coverage

How many levels do you need? About four.

| stack memory capacity | sentences | coverage |
|---|---:|---:|
| no stack memory | 127 | 0.32% |
| 1 stack element | 3,496 | 8.78% |
| 2 stack elements | 25,909 | 65.05% |
| 3 stack elements | 38,902 | 97.67% |
| **4 stack elements** | **39,816** | **99.96%** |
| 5 stack elements | 39,832 | 100.00% |
| TOTAL | 39,832 | 100.00% |

Percent coverage of transformed treebank sections 2–21 w. no punctuation

Good! Because that's supposed to be our limit! (Cowan, 2001)

Now, a windfall in accuracy due to pruned search space?

# Accuracy

No... guessing open adjunction sites to save memory holds back accuracy

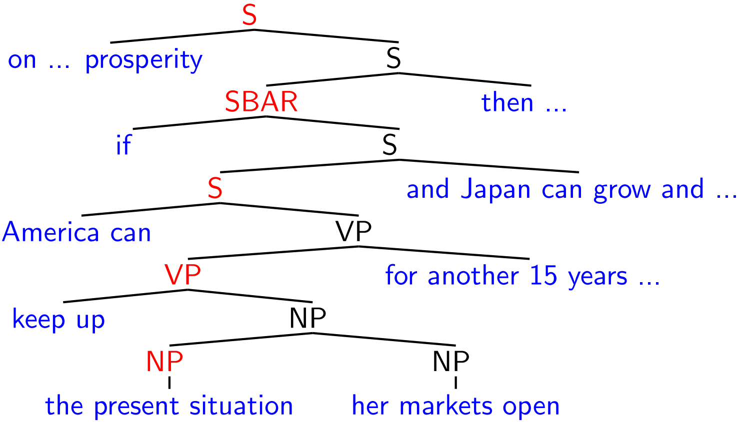
Accuracy results w. no lexicalization or smoothing:

| with punctuation: ($\leq$40 wds) | LP | LR | F | fail |
|---|---|---|---|---|
| KM'03: unmodified, devset | – | – | 72.6 | 0 |
| KM'03: par+sib, devset | – | – | 77.4 | 0 |
| CKY: binarized, devset | 72.3 | 71.1 | 71.7 | 0 |
| **HHMM: par+sib, devset** | **81.4** | **82.9** | **82.1** | **1.4** |
| CKY: binarized, sect 23 | 72.0 | 69.7 | 70.8 | 0.3 |
| **HHMM: par+sib, sect 23** | **79.7** | **80.4** | **80.1** | **0.6** |
| Henderson'04, non-det., sect 0 | | | 89.8 | |

| no punctuation: ($\leq$120 wds) | LP | LR | F | fail |
|---|---|---|---|---|
| R'01: par+sib, sect 23–24 | 77.4 | 75.2 | – | 0.1 |
| **HHMM: par+sib, sect 23–24** | **77.6** | **76.8** | **77.2** | **0.4** |

# Quintuple center-embedding

Here's one of the 16 depth-five sentences in the corpus:




Left-/right-corner won't undo zig-zags. Need them to untangle referents.

# Conclusion

Right-corner transform explains parsing w/in human-like memory limits.

Bounded memory HHMM model mostly safe, in terms of coverage.

But, no big windfall in accuracy.

Future work:

- Lexicalization / vector-space semantics
- Smarter strategy for deferring composition if memory not used up
- Smoothing, backoff
- Estimate joint probabilities over entire columns