Incremental Parsing in Bounded Memory

William Schuler
Department of Linguistics
The Ohio State University

September 16, 2010

Motivation

Goal: simple processing model, matches observations about human memory

1. bounded number of unconnected chunks
   [Miller, 1956, Cowan, 2001]
   (subjects group stimuli into only 4 or so clusters)

Motivation

Goal: simple processing model, matches observations about human memory

1. bounded number of unconnected chunks
   [Miller, 1956, Cowan, 2001]
   (subjects group stimuli into only 4 or so clusters)
2. process rich syntax
   [Chomsky and Miller, 1963]
   (center embedding: 'if [neither [the man [the cop] saw] nor ...] then ...')
   (cf. center recursion: '?? the malt [the rat [the cat chased] ate] ...')

Motivation

Goal: simple processing model, matches observations about human memory

1. bounded number of unconnected chunks
   [Miller, 1956, Cowan, 2001]
   (subjects group stimuli into only 4 or so clusters)
2. process rich syntax
   [Chomsky and Miller, 1963]
   (center embedding: 'if [neither [the man [the cop] saw] nor ...] then ...')
   (cf. center recursion: '?? the malt [the rat [the cat chased] ate] ...')
3. processing is incremental
   [Sachs, 1967, Jarvella, 1971]
   (subjects can't remember specific words earlier in sentence)

Motivation

Goal: simple processing model, matches observations about human memory

1. bounded number of unconnected chunks
   [Miller, 1956, Cowan, 2001]
   (subjects group stimuli into only 4 or so clusters)
2. process rich syntax
   [Chomsky and Miller, 1963]
   (center embedding: 'if [neither [the man [the cop] saw] nor ...] then ...')
   (cf. center recursion: '?? the malt [the rat [the cat chased] ate] ...')
3. processing is incremental
   [Sachs, 1967, Jarvella, 1971]
   (subjects can't remember specific words earlier in sentence)
4. processing is parallel, probabilistic
   [Jurafsky, 1996, Hale, 2001, Levy, 2008]
   (probabilistic / info. theoretic measures correlate w. reading times)

Motivation

Goal: simple processing model, matches observations about human memory

In particular, we use a factored sequence model (dynamic Bayes net):

1. random variables in Bayesian model are easily interpretable
   (explicit estimation of speaker intent; cf. neural net)

Motivation

Goal: simple processing model, matches observations about human memory

In particular, we use a factored sequence model (dynamic Bayes net):

1. random variables in Bayesian model are easily interpretable
   (explicit estimation of speaker intent; cf. neural net)
2. clear role of bounded working memory store
   (random variable for each store element)

Motivation

Goal: simple processing model, matches observations about human memory

In particular, we use a factored sequence model (dynamic Bayes net):

1. random variables in Bayesian model are easily interpretable
   (explicit estimation of speaker intent; cf. neural net)
2. clear role of bounded working memory store
   (random variable for each store element)
3. clear role of syntax
   (grammar transform turns trees into chunks for store elements)

Motivation

Goal: simple processing model, matches observations about human memory

In particular, we use a factored sequence model (dynamic Bayes net):

1. random variables in Bayesian model are easily interpretable
   (explicit estimation of speaker intent; cf. neural net)
2. clear role of bounded working memory store
   (random variable for each store element)
3. clear role of syntax
   (grammar transform turns trees into chunks for store elements)
4. fast enough to interact with real-time speech recognizer
   (using cool engineering tricks: best-first / 'lazy' k-best search)

Motivation

Goal: simple processing model, matches observations about human memory

In particular, we use a factored sequence model (dynamic Bayes net):

1. random variables in Bayesian model are easily interpretable
   (explicit estimation of speaker intent; cf. neural net)
2. clear role of bounded working memory store
   (random variable for each store element)
3. clear role of syntax
   (grammar transform turns trees into chunks for store elements)
4. fast enough to interact with real-time speech recognizer
   (using cool engineering tricks: best-first / 'lazy' k-best search)

Result is a nice platform for linguistic experimentation!

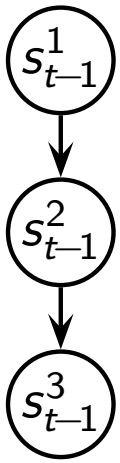
Overview

Tutorial talk:

- Part I: Incremental Parsing
  - bounded-memory sequence model
  - connection to phrase structure
  - coverage
  - implementation/evaluation as performance model
- Part II: Extensions (Semantic Dependencies)
  - preserving probabilistic dependencies in sequence model
  - preserving semantic dependencies in sequence model
  - interactive speech interpretation
  - an analysis of non-local dependencies

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine

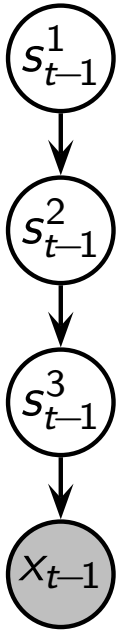


DBN: circles=random variables (mem store elements), arcs=dependencies

Elements hold hypoth. stacked-up **incomplete constituents**, dep. on parent (incomplete constituent: e.g. S/VP = sentence lacking verb phrase to come)

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine



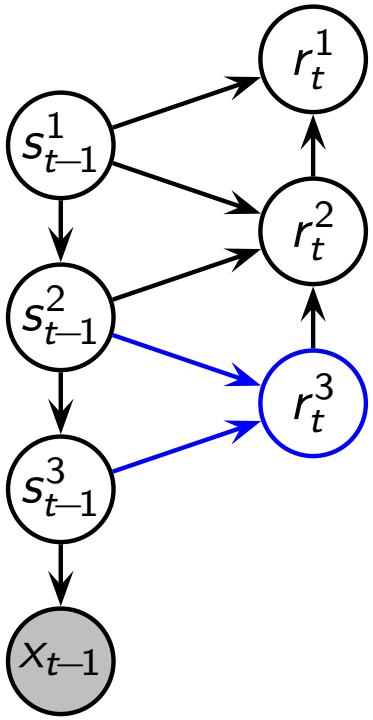
DBN: circles=random variables (mem store elements), arcs=dependencies

Elements hold hypoth. stacked-up **incomplete constituents**, dep. on parent (incomplete constituent: e.g. S/VP = sentence lacking verb phrase to come)

Hypothesized mem elements generate **observations**: words / acoust. features

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine

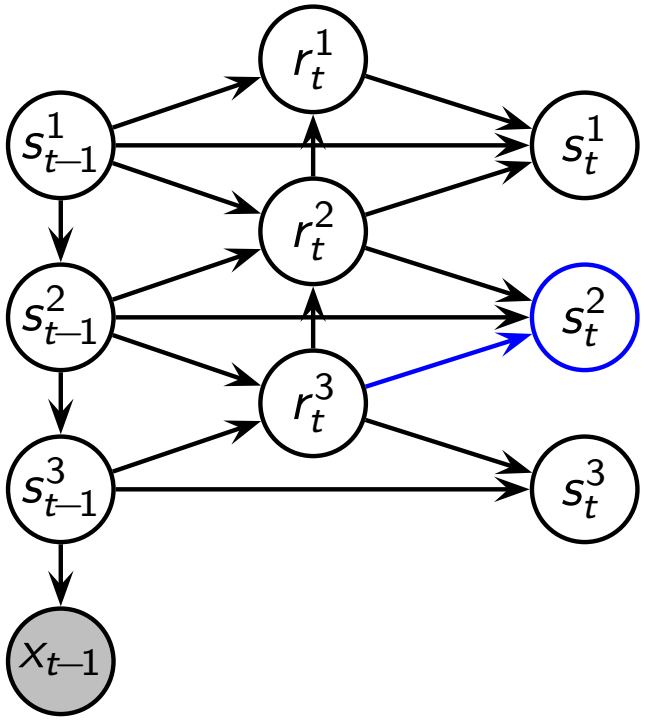


Elements in memory store may be composed (reduced) w. element above

Probability depends on dependent vars (e.g. Det, Noun reduce to NP)

Probabilistic Sequence Model

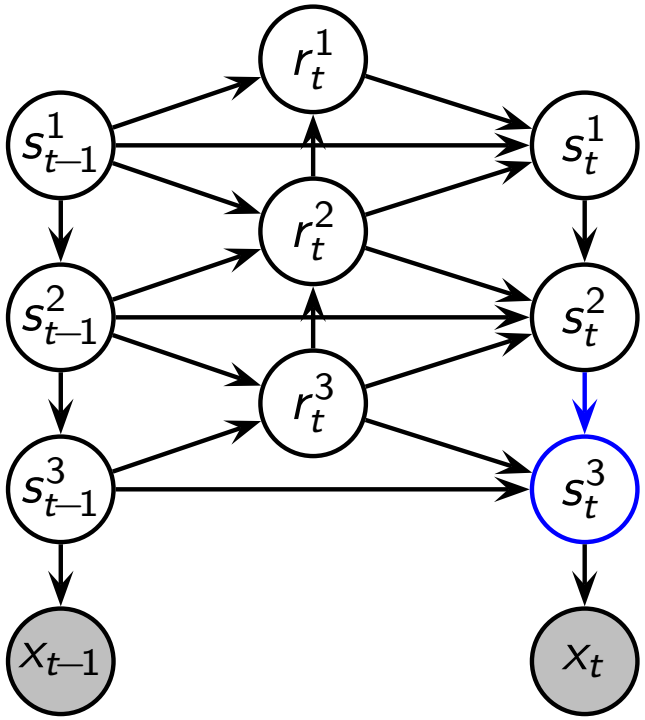
Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine



(Non-)reduced elements carry forward or transition (e.g. NP becomes S/VP)

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine

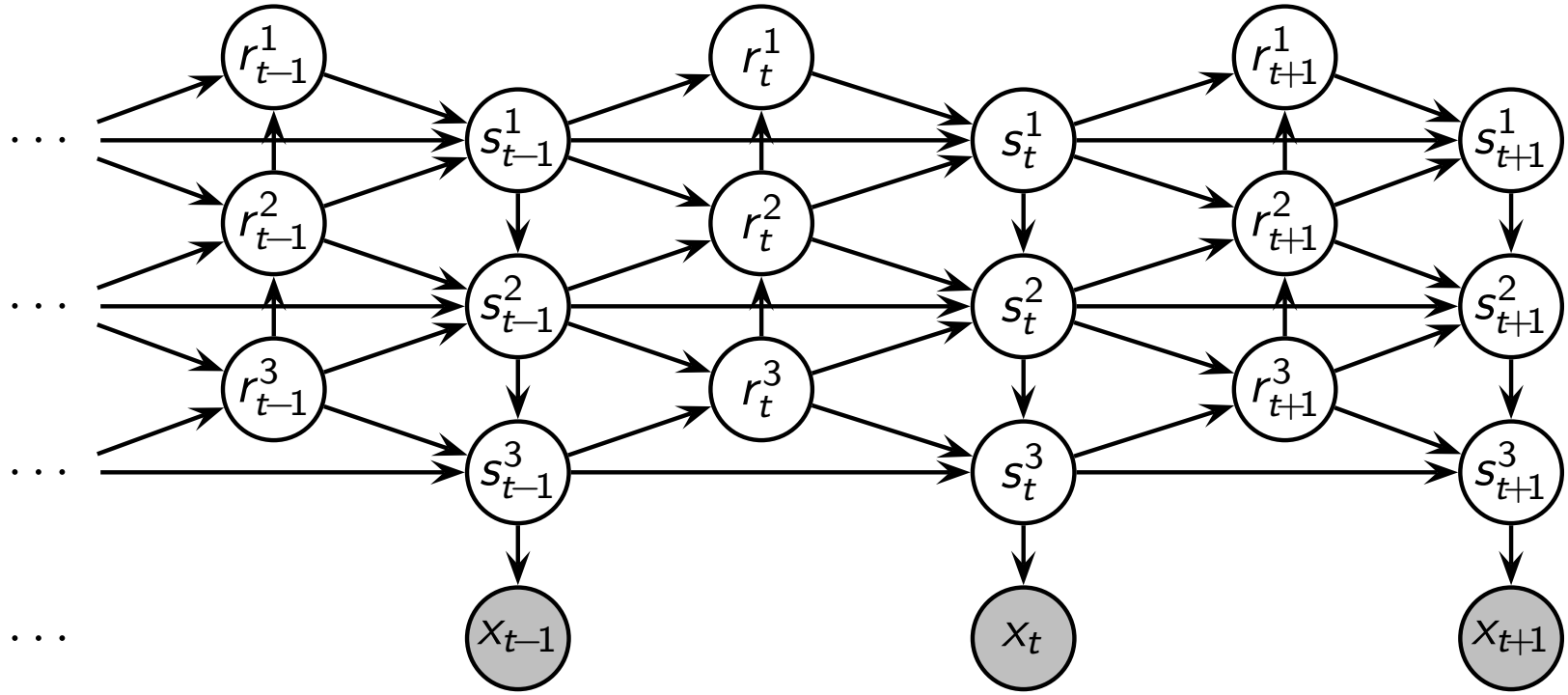


(Non-)reduced elements carry forward or transition (e.g. NP becomes S/VP)

Transitioned elements may be expanded again (e.g. S/VP expands to Verb)

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine



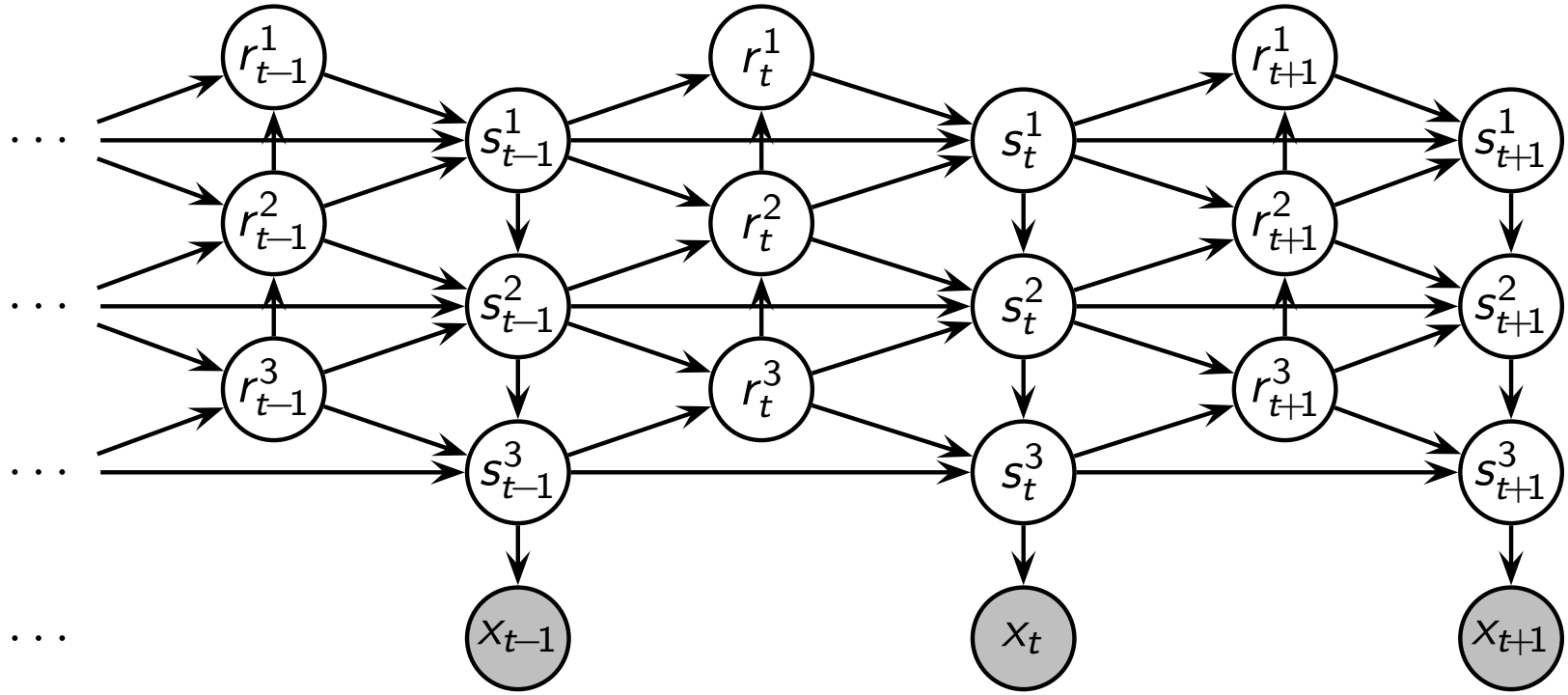
(Non-)reduced elements carry forward or transition (e.g. NP becomes S/VP)

Transitioned elements may be expanded again (e.g. S/VP expands to Verb)

Process continues through time

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine

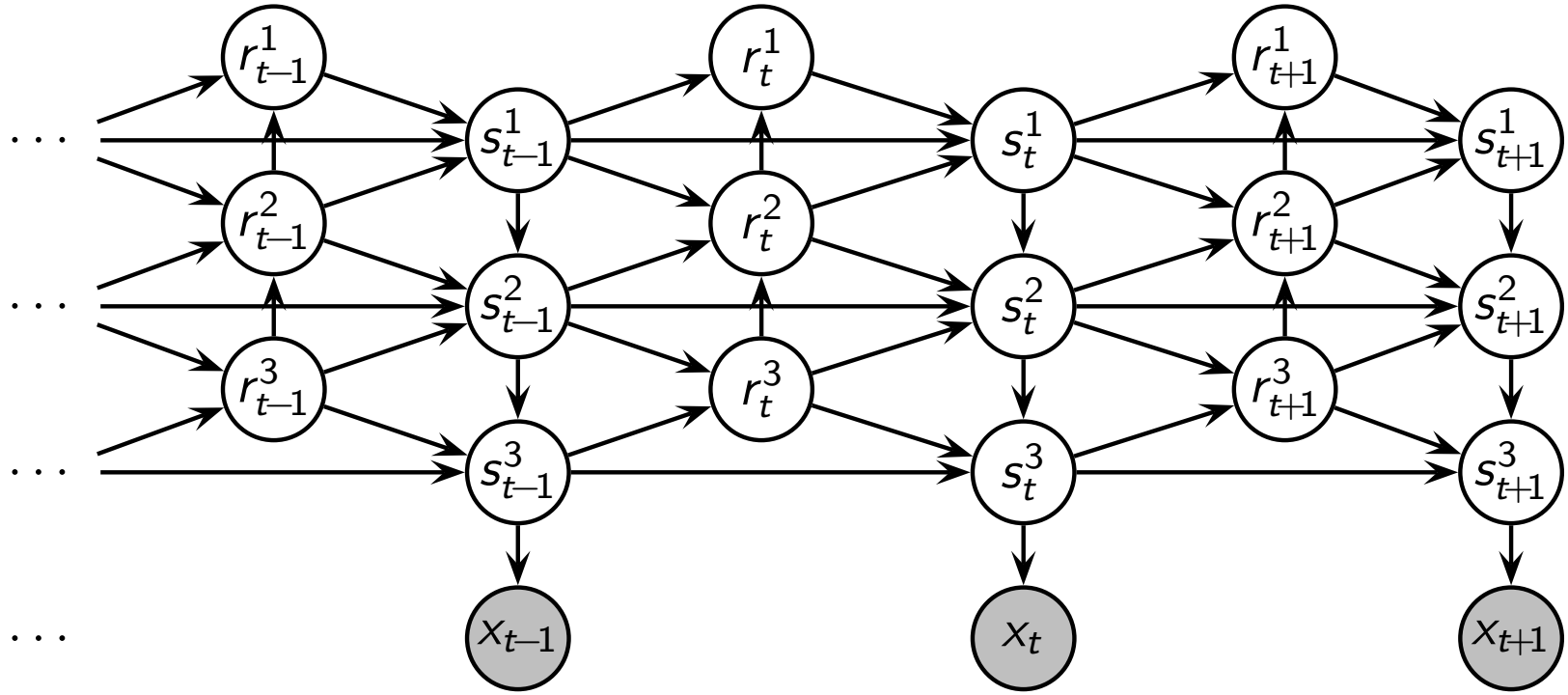


Alternate hypotheses (memory store configurations) compete w. each other:

$$\hat{s}_{1..T}^{1..D} \stackrel{\text{def}}{=} \operatorname*{argmax}_{s_{1..T}^{1..D}} \prod_{t=1}^{T} \mathrm{P}_{\theta_Y}(s_t^{1..D} \mid s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_X}(x_t \mid s_t^{1..D})$$

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine

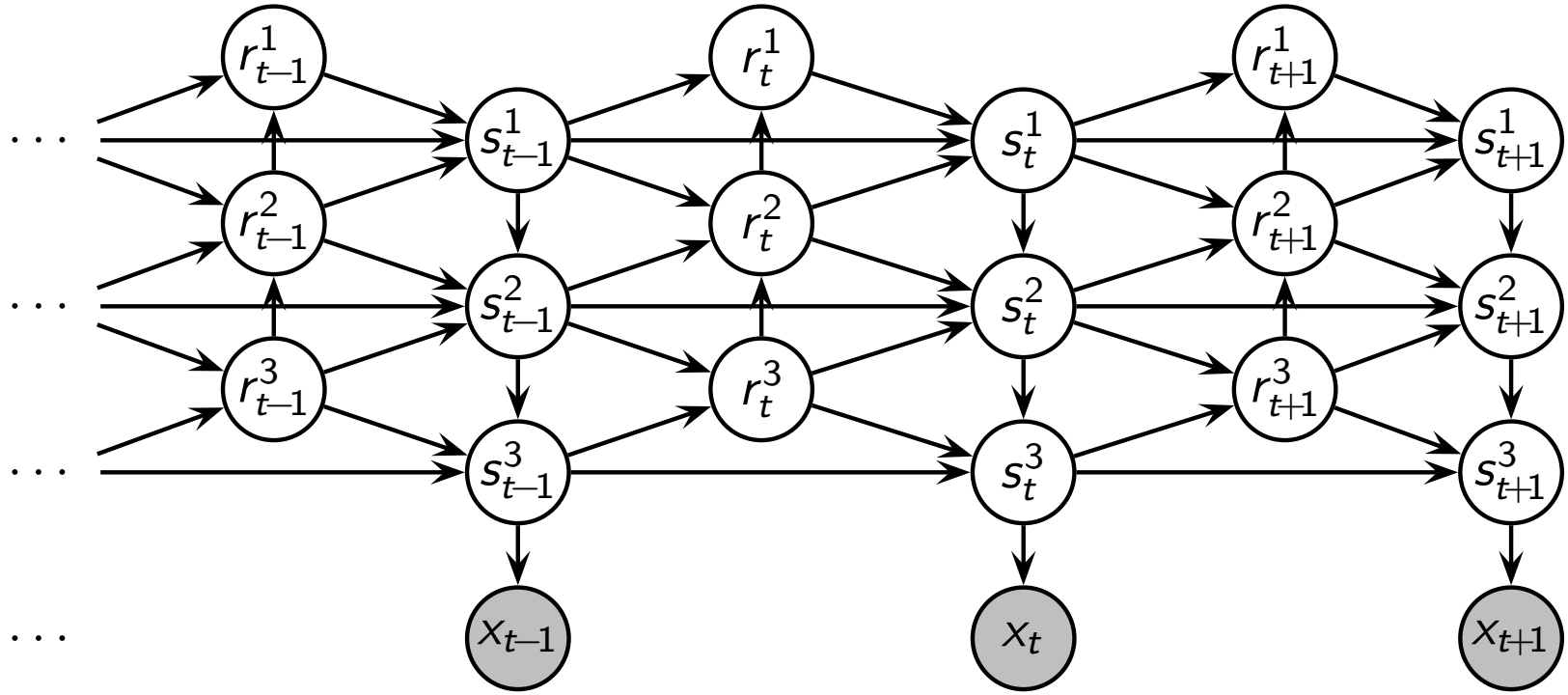


$$\mathrm{P}_{\theta_Y}(s_t^{1..D} \mid s_{t\text{-}1}^{1..D}) = \sum_{r_t^{1..D}} \mathrm{P}_{\theta_{\text{Reduce}}}(r_t^{1..D} \mid s_{t\text{-}1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{Shift}}}(s_t^{1..D} \mid r_t^{1..D}\ s_{t\text{-}1}^{1..D})$$

$$\stackrel{\text{def}}{=} \sum_{r_t^{1..D}} \prod_{d=1}^{D} \mathrm{P}_{\theta_{\mathrm{R},d}}(r_t^d \mid r_t^{d+1} s_{t\text{-}1}^d s_{t\text{-}1}^{d\text{-}1}) \cdot \mathrm{P}_{\theta_{\mathrm{S},d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t\text{-}1}^d s_t^{d\text{-}1})$$

Probabilistic Sequence Model

Hierarch. Hidden Markov Model [Murphy,Paskin'01]: bounded stack machine

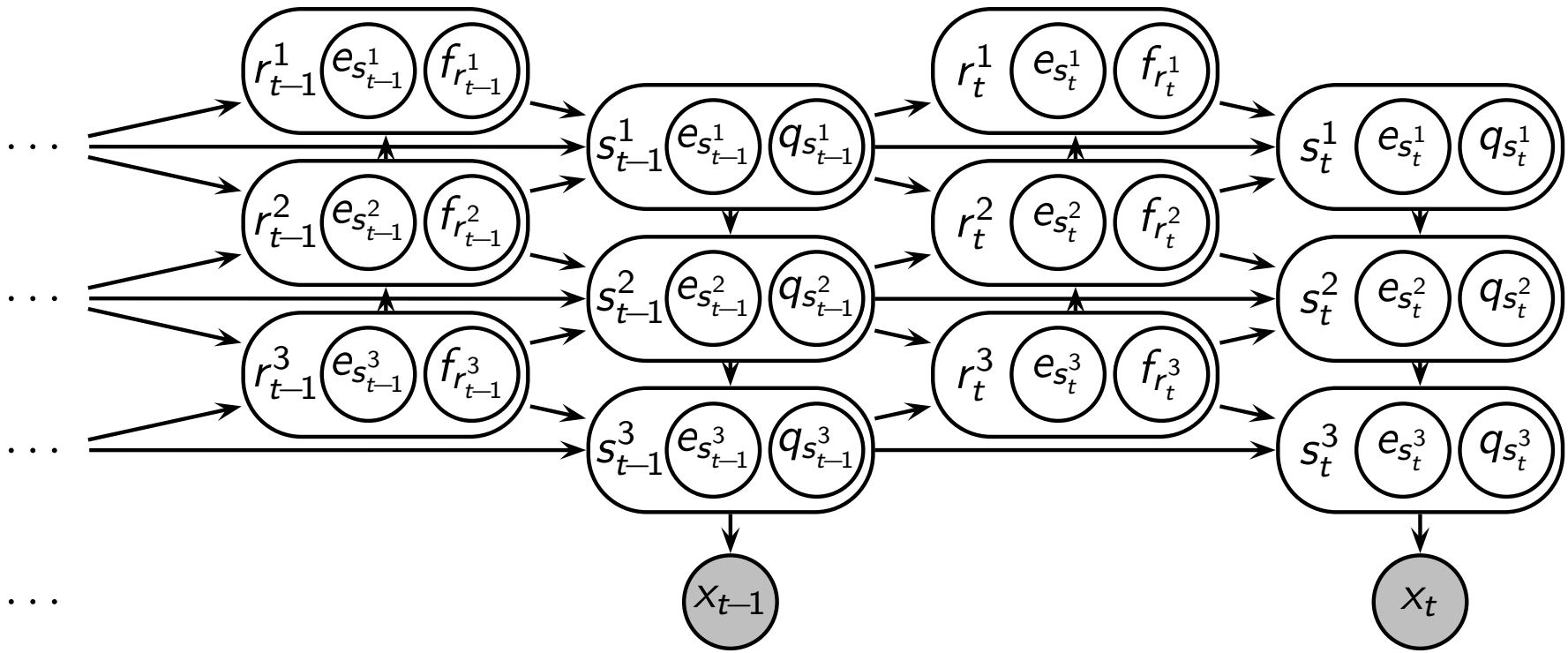


Natural independence assumptions:

- time-order: each store depends on immediately previous store
- depth-order: incomplete constituents separated by yet-unknown struct

Probabilistic Sequence Model

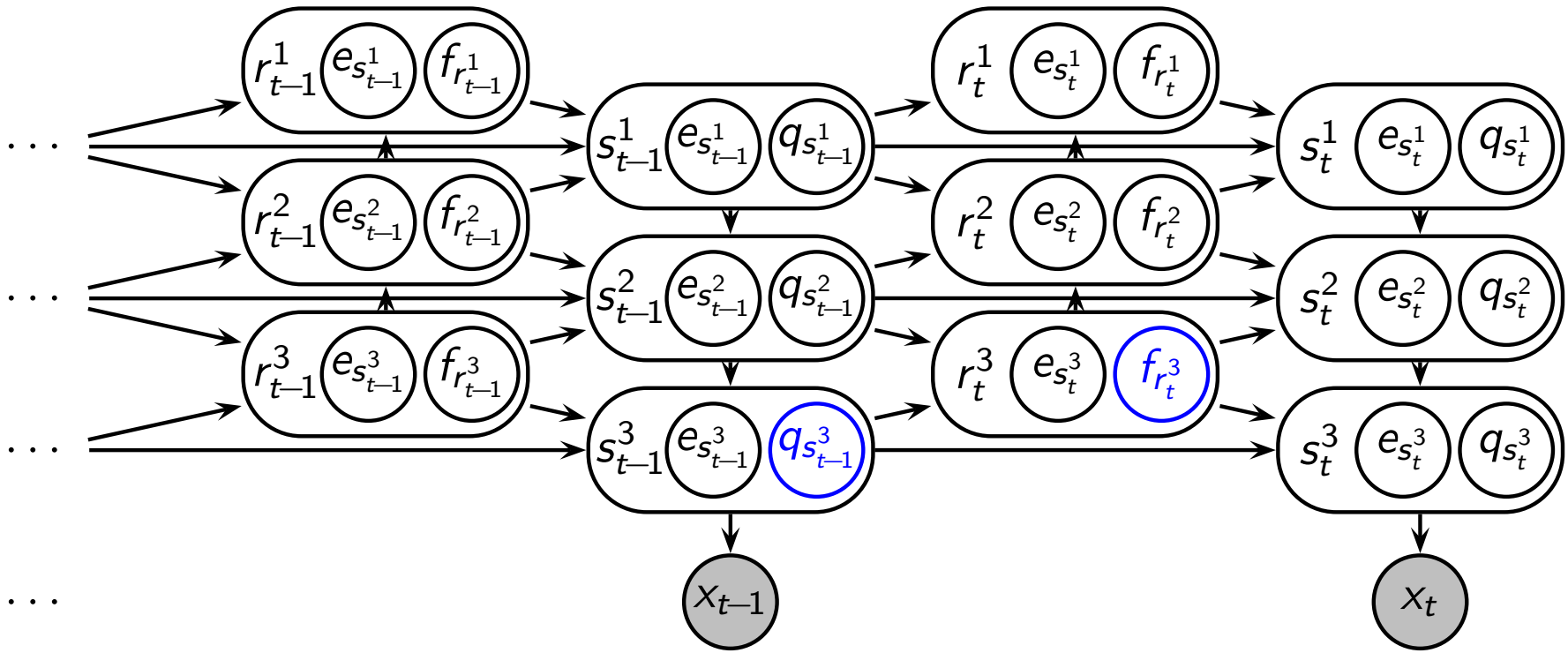
Add interactive semantics — simply factor HHMM states:



— factor r, s into interdependent syntactic (q/f) and referential (e) states:

Probabilistic Sequence Model

Add interactive semantics — simply factor HHMM states:

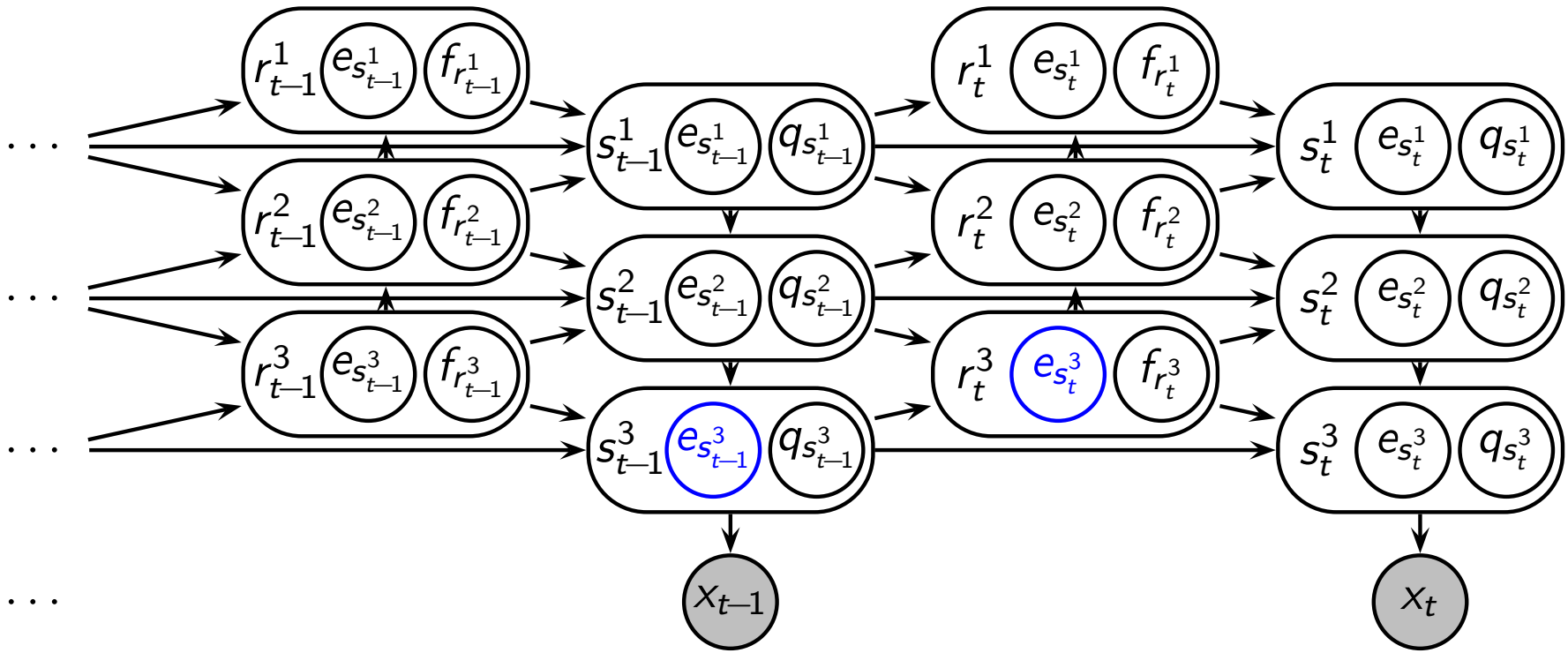


— factor r, s into interdependent syntactic (q/f) and referential (e) states:

- **incomplete syntactic states:** e.g. $q = \mathrm{S/VP}$ (with f_r as a reduce flag)

Probabilistic Sequence Model

Add interactive semantics — simply factor HHMM states:



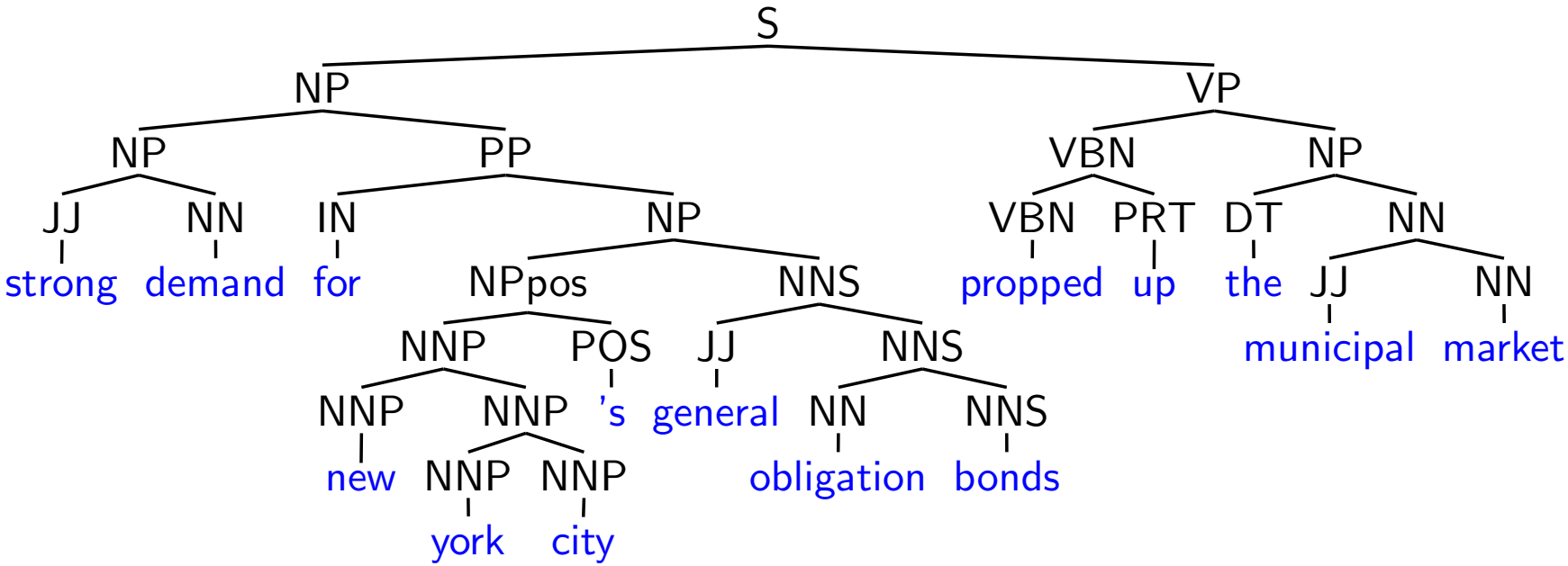
— factor r, s into interdependent syntactic (q/f) and referential (e) states:

- **incomplete syntactic states:** e.g. $q = \mathrm{S/VP}$ (with f_r as a reduce flag)
- **incomplete referential states:** e.g. $e = \{i_{\text{coling}}, i_{\text{naacl}}\}$ (concept set/reln)

Connecting Phrase Structure to Sequence Model

Sequences of memory stores correspond directly to familiar phrase structure:

(trees from Penn Treebank, binarized around head rules)



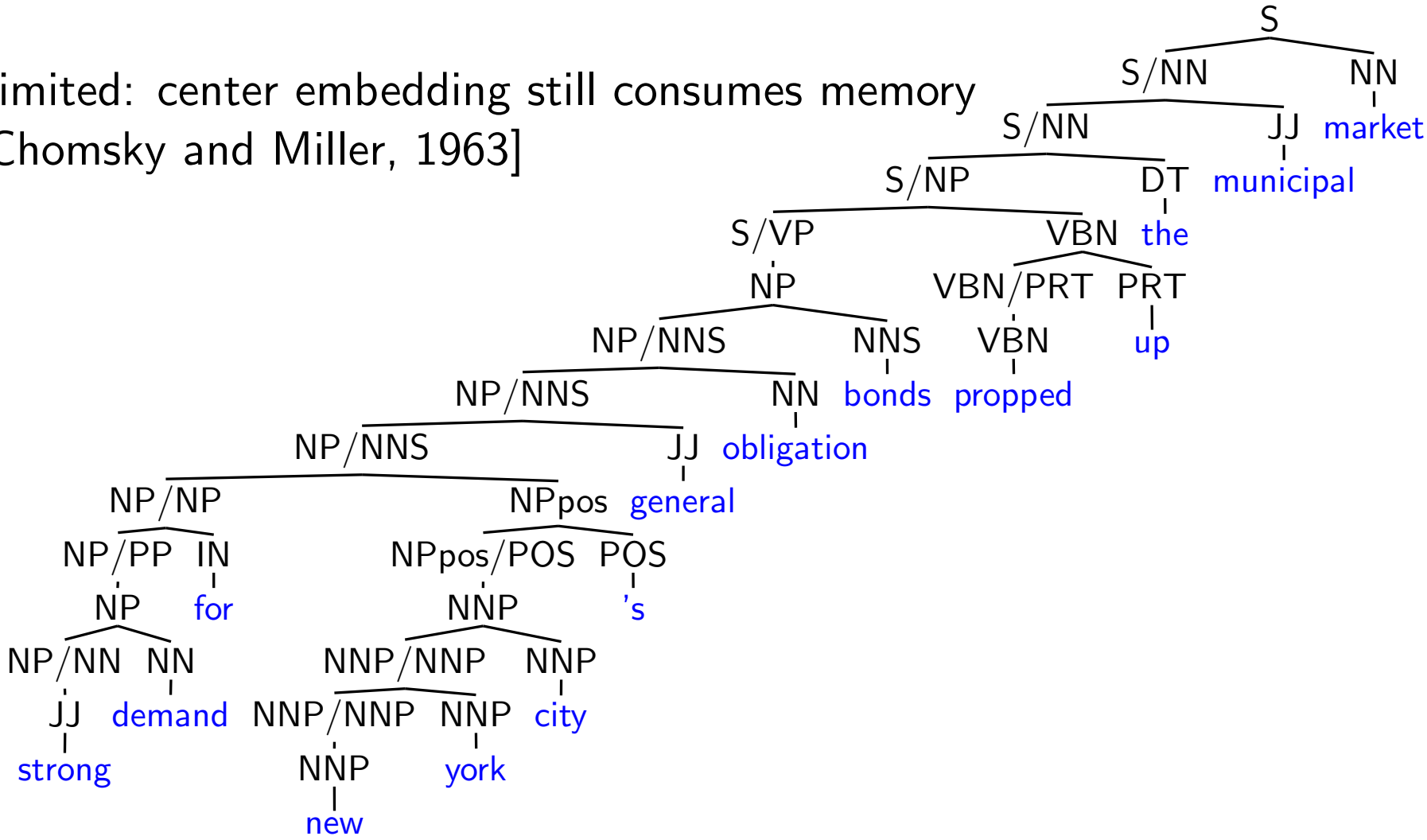
Phrase structure allows nested expressions: 'prop ... up,' 'if ... then ...'

Bounded memory requires flatter, more memory-efficient representation...

Connecting Phrase Structure to Sequence Model

'Right-corner transform' map right-embedded sequence → left-embedded seq.
(allows new constituents to be immediately composed)

Limited: center embedding still consumes memory
[Chomsky and Miller, 1963]

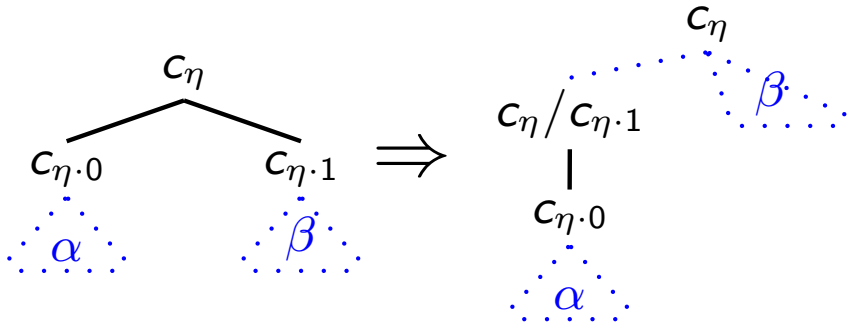


Connecting Phrase Structure to Sequence Model

Transform is simple — **three cases** on any right-embedded sequence:

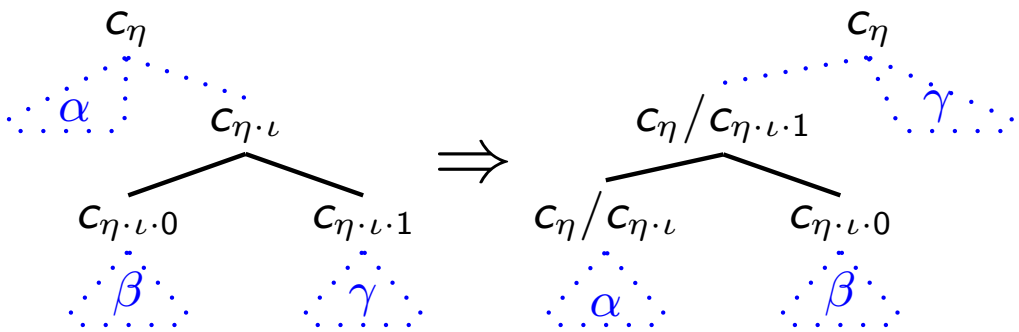
(η, ι are paths of 0:left/1:right)

Beginning case:



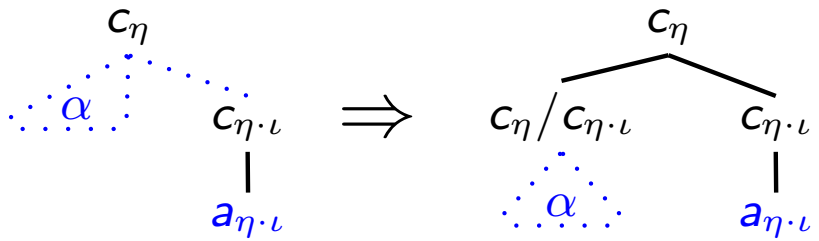
(∼ CCG type raising)

Middle case:



(∼ fwd. composition)

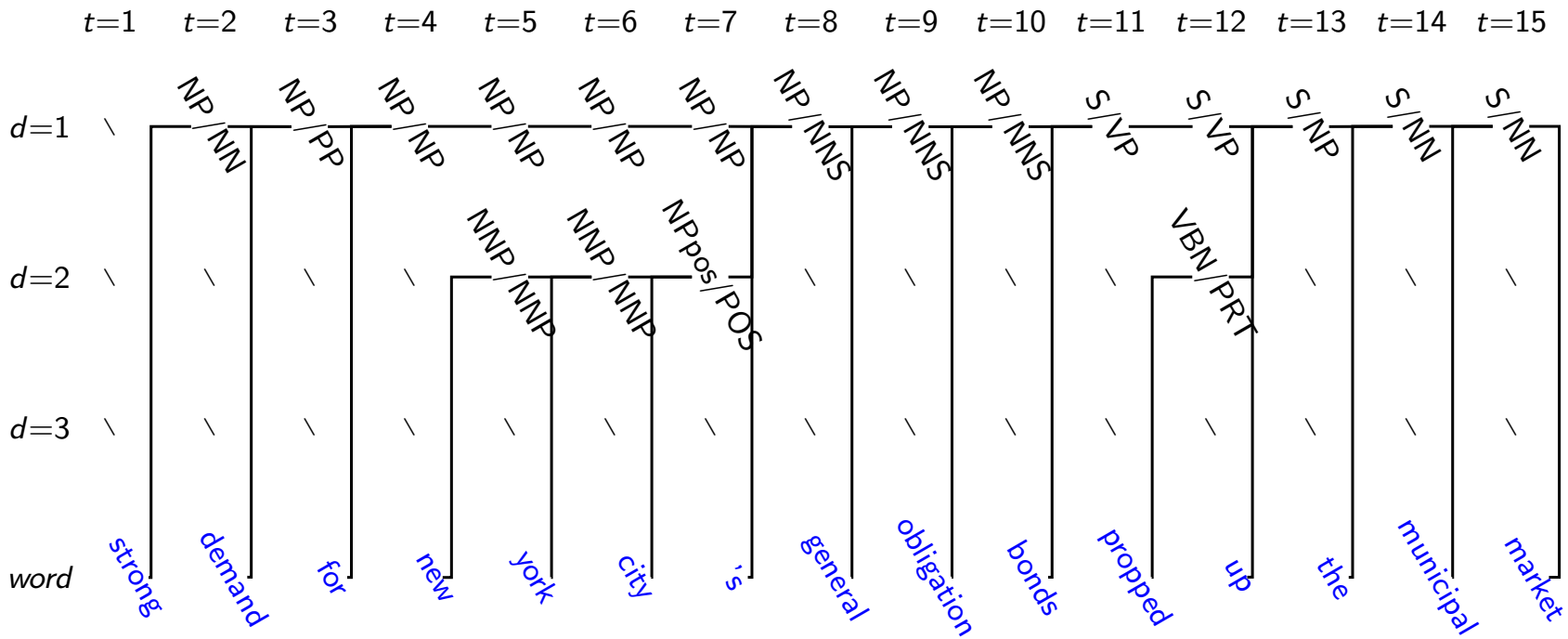
Ending case:



(∼ fwd. application)

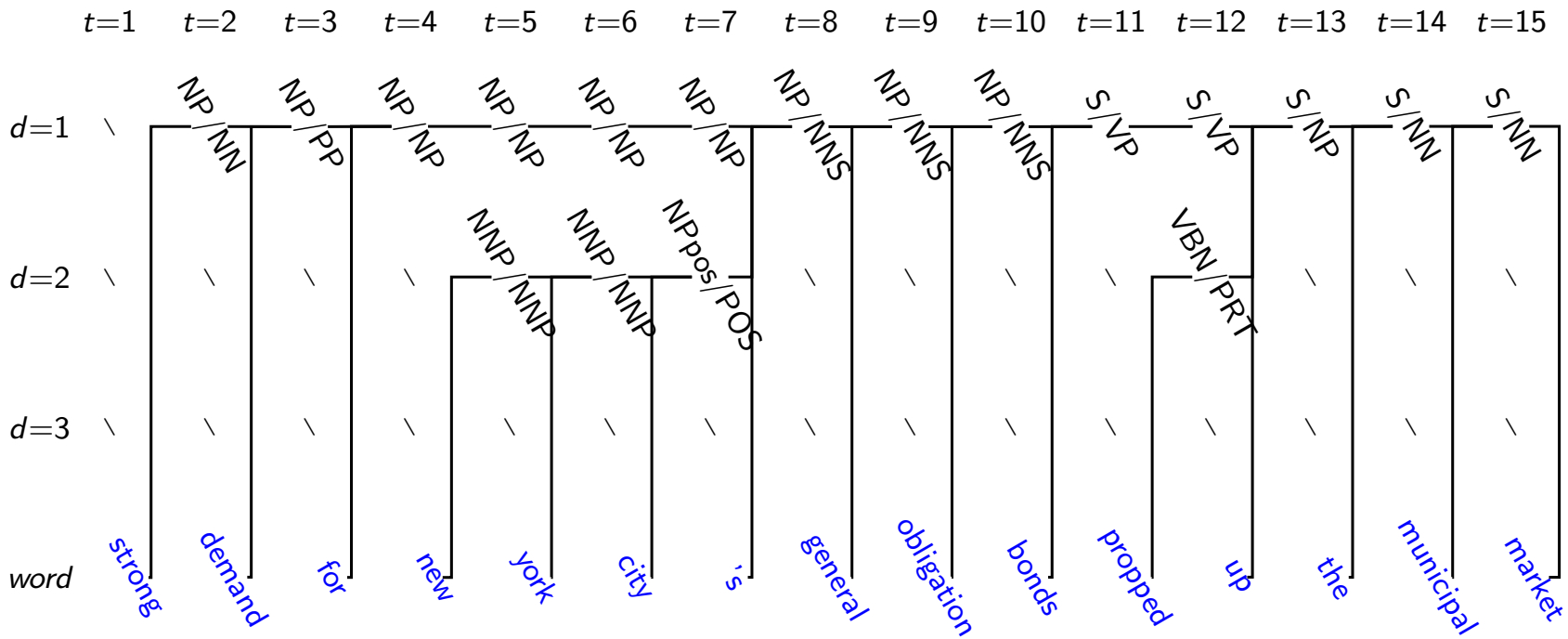
Connecting Phrase Structure to Sequence Model

Align trees to HHMM grid, train variables on incomplete constit. 'chunks':



Connecting Phrase Structure to Sequence Model

Align trees to HHMM grid, train variables on incomplete constit. 'chunks':

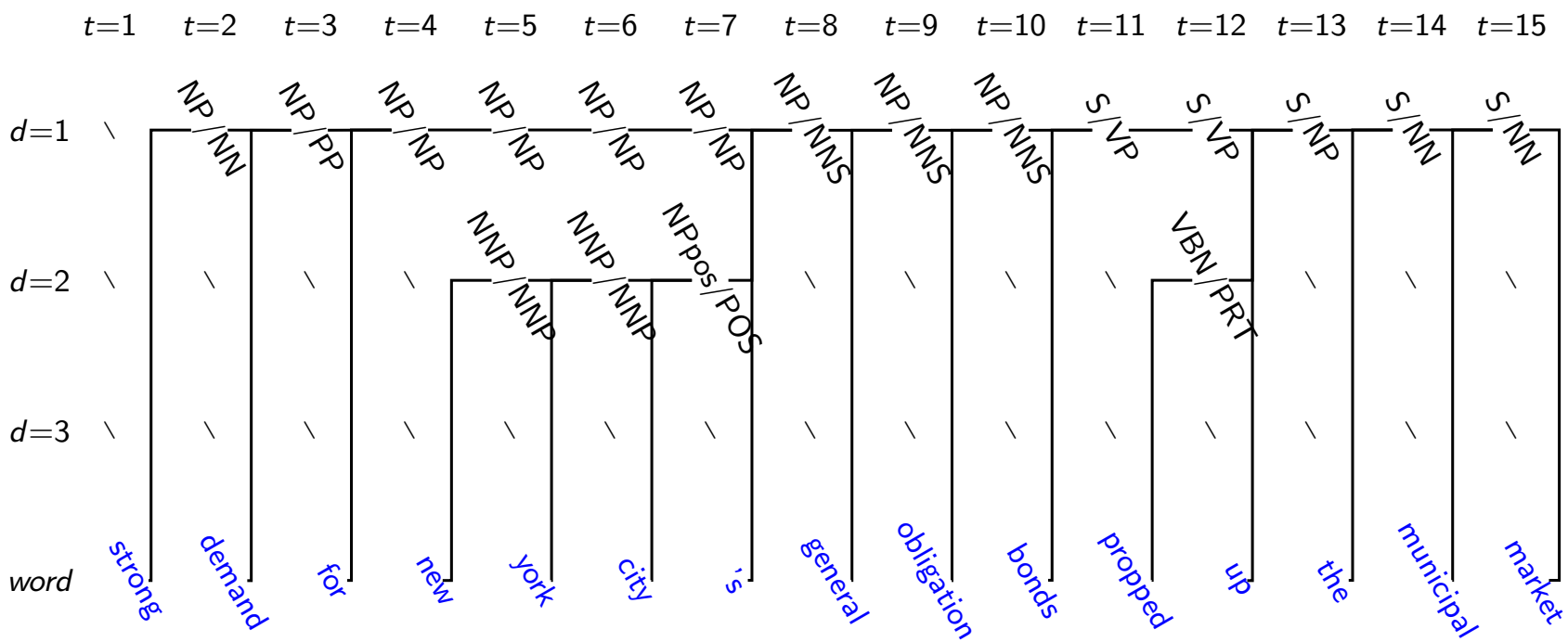


Unlike incomplete constit. chunks from left-corner transform [Johnson '98]:

- right-corner gives fixed struct w/in chunk, unknown struct between
- time-order traversal gives bottom-up (compositional) semantics

Connecting Phrase Structure to Sequence Model

Align trees to HHMM grid, train variables on incomplete constit. 'chunks':



Unlike full Combinatory Categorial Grammar [Steedman, 2000]:

- not pure lexical: semantics can come from grammar rules/'constructions'
- no commitment to set of combinators

Coverage

Penn Treebank WSJ coverage experiment indicates 3- to 4-element store [Schuler et al., 2008, Schuler et al., 2010]

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/o punctuation)
Good, because 3 to 4 elements supposed to be our limit [Cowan, 2001]

Coverage

Penn Treebank WSJ coverage experiment indicates 3- to 4-element store [Schuler et al., 2008, Schuler et al., 2010]

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%

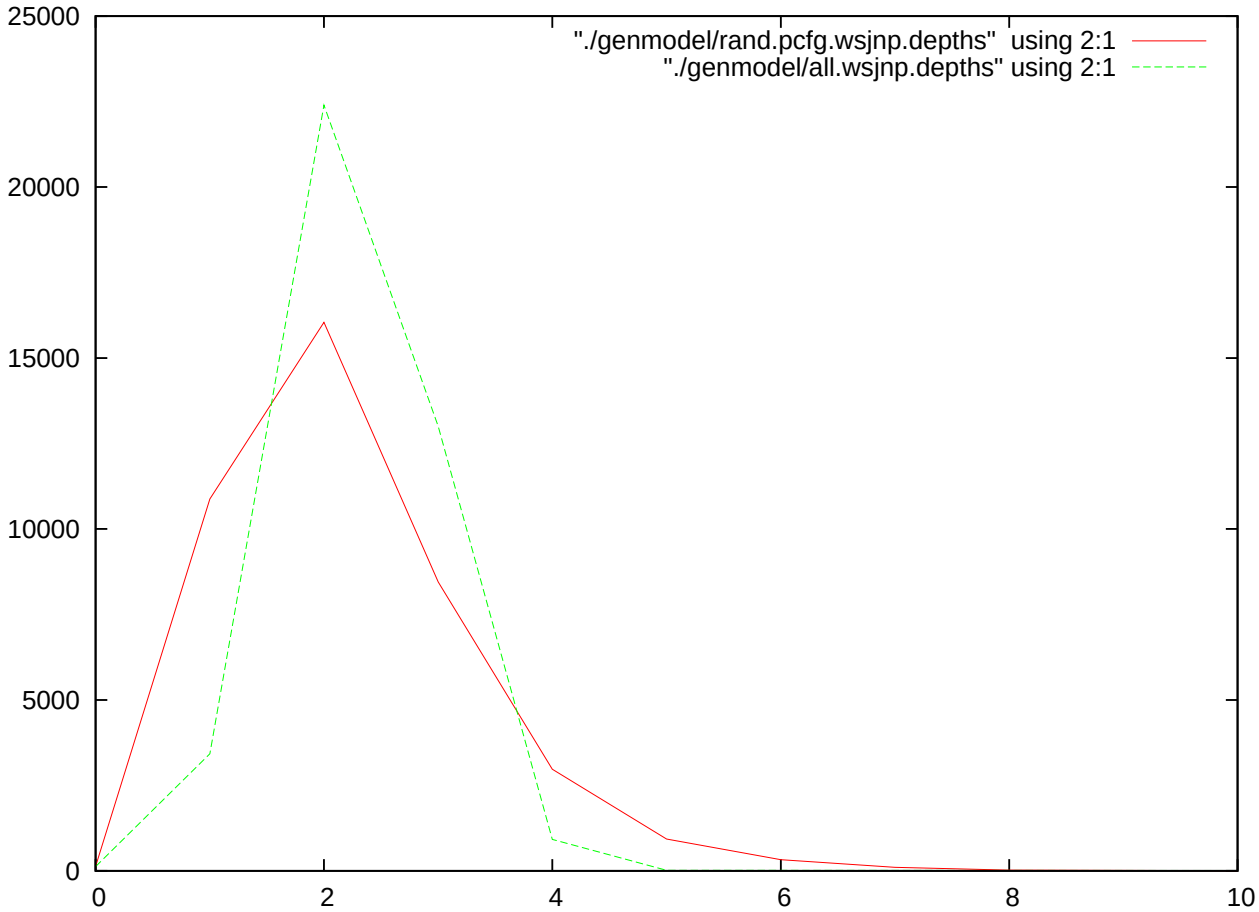
(percent coverage of transformed Treebank sections 2–21 w/o punctuation)
Good, because 3 to 4 elements supposed to be our limit [Cowan, 2001]

Stronger for Switchboard (spontaneous speech): 92.1% at 2, 99.5% at 3.

Coverage

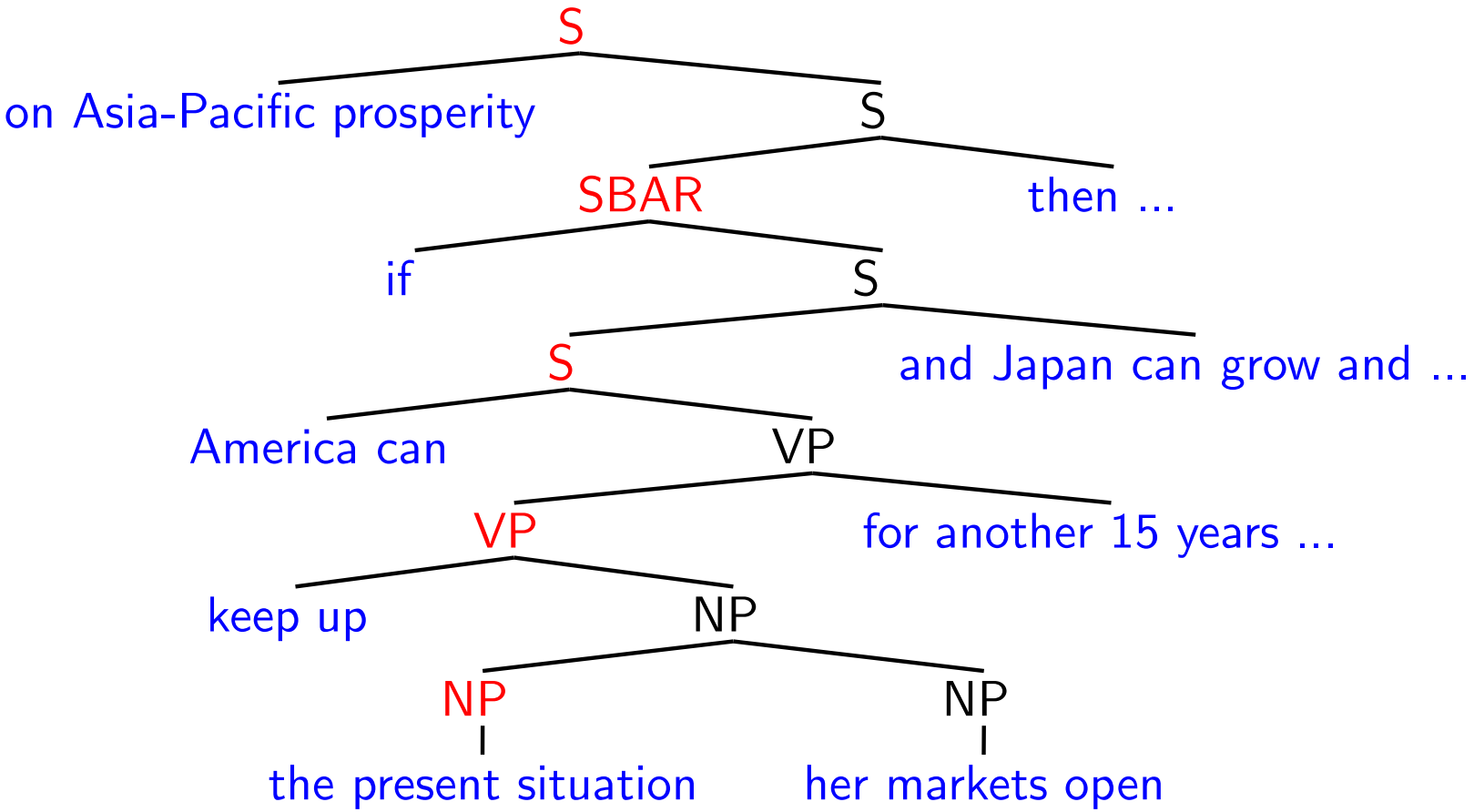
Penn Treebank WSJ coverage experiment indicates 3- to 4-element store

Significant ($p<.0001$) divergence from corpus randomly generated from pcfg



Coverage

Here's one of the 16 depth-five sentences in the (40,000 sent.) corpus:



Does indeed seem like a lot to remember

Evaluation as a Performance Model

Setting D=4 should allow nearly complete coverage.

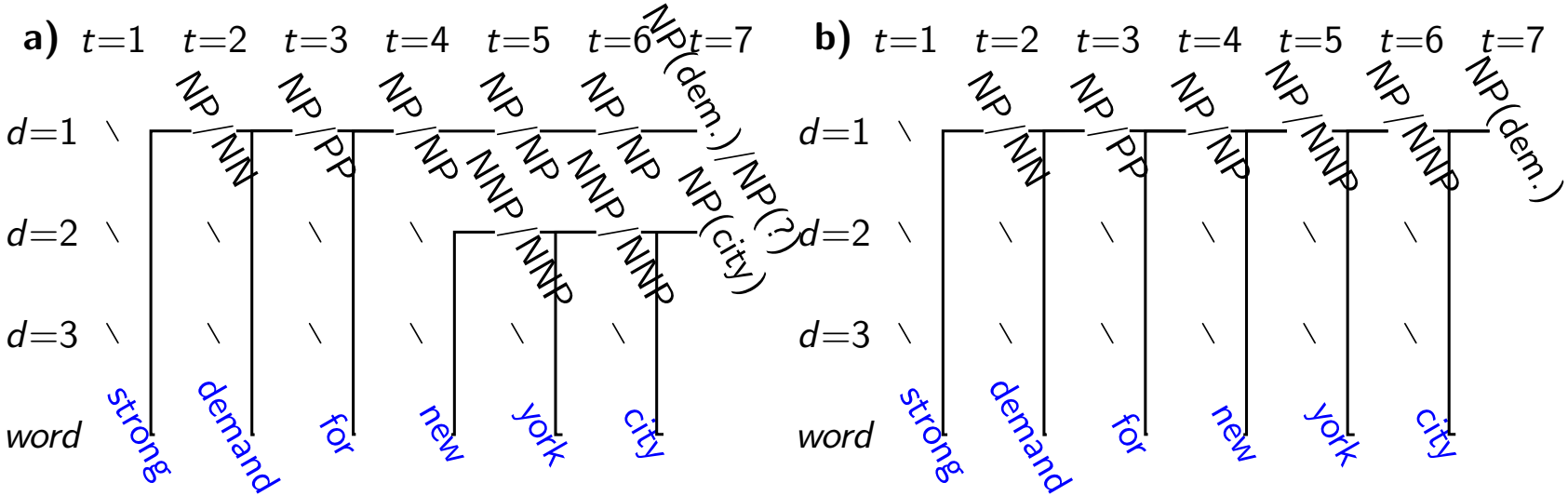
But strict memory bounds may introduce spurious local ambiguity...

Evaluation as a Performance Model

Setting D=4 should allow nearly complete coverage.

But strict memory bounds may introduce spurious local ambiguity...

Bounded memory forces model to anticipate nesting demands:
(a) allow modification to 'city' NP / (b) leave room for future embedding:

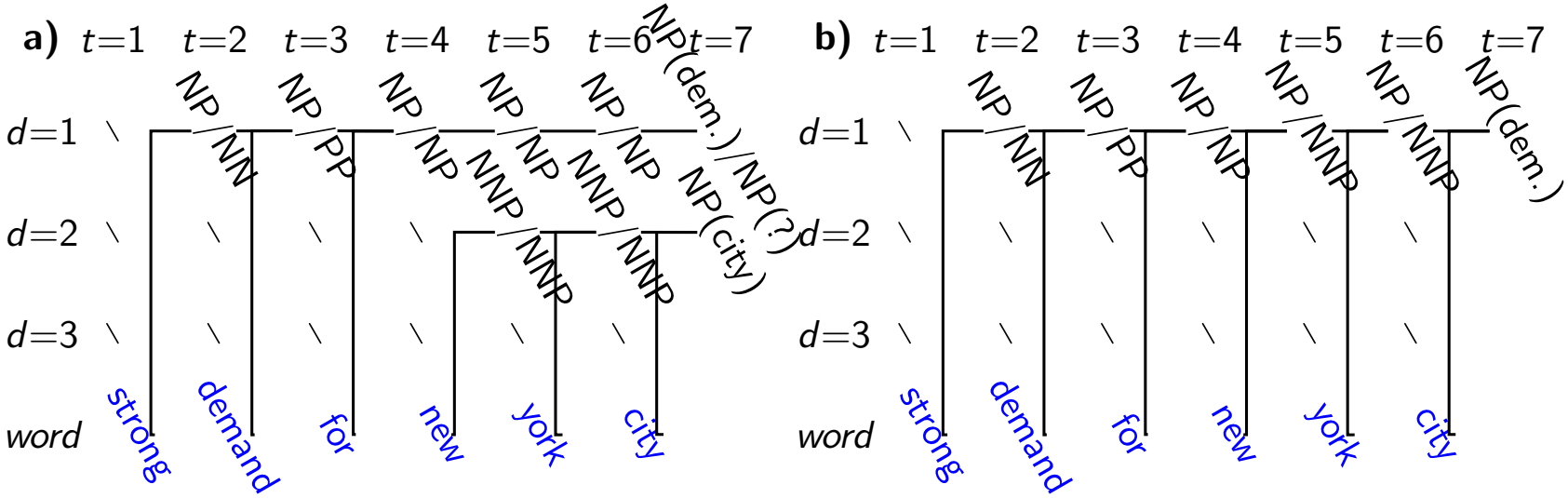


Evaluation as a Performance Model

Setting D=4 should allow nearly complete coverage.

But strict memory bounds may introduce spurious local ambiguity...

Bounded memory forces model to anticipate nesting demands:
(a) allow modification to 'city' NP / (b) leave room for future embedding:



As defined, model makes this prediction probabilistically
(parsing strategy is optionally 'arc-eager' [Abney and Johnson, 1991]).
But, could be overwhelmed with memory-management ambiguity...

Evaluation as a Performance Model

Implemented using 'Modelblocks' C++ templates for sequence models:

$$\mathrm{P}_{\theta_Y}(s_t^{1..D} \mid s_{t-1}^{1..D}) = \sum_{r_t^{1..D}} \mathrm{P}_{\theta_{\text{Reduce}}}(r_t^{1..D} \mid s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{Shift}}}(s_t^{1..D} \mid r_t^{1..D}\ s_{t-1}^{1..D})$$

$$\stackrel{\text{def}}{=} \sum_{r_t^{1..D}} \prod_{d=1}^{D} \mathrm{P}_{\theta_{\mathrm{R},d}}(r_t^d \mid r_t^{d+1} s_{t-1}^d s_{t-1}^{d-1}) \cdot \mathrm{P}_{\theta_{\mathrm{S},d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t-1}^d s_t^{d-1})$$

```
class YModel {
 public:
  RModel mR;
  SModel mS;
  LogProb setIterProb ( Y::AIterator<LogProb>& y, const S& sP, const X& x, bool b1, int& a ) const {
    LogProb pr;
    pr  = mR.setIterProb ( y.first, sP, b1, a );
    pr *= mS.setIterProb ( y.second, y.first, sP, a );
    return pr;
  }
  // ...
};
```

Evaluation as a Performance Model

Implemented using 'Modelblocks' C++ templates for sequence models:

$$\mathrm{P}_{\theta_{\mathrm{Y}}}(s_t^{1..D} \mid s_{t\text{-}1}^{1..D}) = \sum_{r_t^{1..D}} \mathrm{P}_{\theta_{\mathrm{Reduce}}}(r_t^{1..D} \mid s_{t\text{-}1}^{1..D}) \cdot \mathrm{P}_{\theta_{\mathrm{Shift}}}(s_t^{1..D} \mid r_t^{1..D}\ s_{t\text{-}1}^{1..D})$$

$$\stackrel{\text{def}}{=} \sum_{r_t^{1..D}} \prod_{d=1}^{D} \mathrm{P}_{\theta_{\mathrm{R},d}}(r_t^d \mid r_t^{d+1} s_{t\text{-}1}^d s_{t\text{-}1}^{d\text{-}1}) \cdot \mathrm{P}_{\theta_{\mathrm{S},d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t\text{-}1}^d s_t^{d\text{-}1})$$

```
class RModel {
 public:
  RdModel mRd;
  LogProb setIterProb ( R::AIterator<LogProb>& r, const S& sP, bool b1, int& a ) const {
    LogProb pr;
    pr  = mRd.setIterProb ( r[3], 4, Rd(sP.second), sP.first[3], sP.first[2], b1, a );
    pr *= mRd.setIterProb ( r[2], 3, Rd(r[3]),      sP.first[2], sP.first[1], b1, a );
    pr *= mRd.setIterProb ( r[1], 2, Rd(r[2]),      sP.first[1], sP.first[0], b1, a );
    pr *= mRd.setIterProb ( r[0], 1, Rd(r[1]),      sP.first[0], Sd_TOP,      b1, a );
    return pr;
  }
};
```

Evaluation as a Performance Model

Implemented using 'Modelblocks' C++ templates for sequence models:

$$\mathrm{P}_{\theta_Y}(s_t^{1..D} \mid s_{t-1}^{1..D}) = \sum_{r_t^{1..D}} \mathrm{P}_{\theta_{\text{Reduce}}}(r_t^{1..D} \mid s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{Shift}}}(s_t^{1..D} \mid r_t^{1..D}\ s_{t-1}^{1..D})$$

$$\stackrel{\text{def}}{=} \sum_{r_t^{1..D}} \prod_{d=1}^{D} \mathrm{P}_{\theta_{\mathrm{R},d}}(r_t^d \mid r_t^{d+1} s_{t-1}^d s_{t-1}^{d-1}) \cdot \mathrm{P}_{\theta_{\mathrm{S},d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t-1}^d s_t^{d-1})$$

```
class SModel {
 public:
  SdModel mSd;
  LogProb setIterProb(S::AIterator<LogProb>& s, const R::AIterator<LogProb>& r, const S& sP, int& a) const {
    LogProb pr;
    pr  = mSd.setIterProb ( s.first[0], 1, Rd(r[1]),  Rd(r[0]), sP.first[0], Sd_TOP,          a );
    pr *= mSd.setIterProb ( s.first[1], 2, Rd(r[2]),  Rd(r[1]), sP.first[1], Sd(s.first[0]), a );
    pr *= mSd.setIterProb ( s.first[2], 3, Rd(r[3]),  Rd(r[2]), sP.first[2], Sd(s.first[1]), a );
    pr *= mSd.setIterProb ( s.first[3], 4, sP.second, Rd(r[3]), sP.first[3], Sd(s.first[2]), a );
    pr *= ( G(s.first[3].second)!=G_BOT && G(s.first[3].second).getTerm()!=B_1 )
      ?  mSd.mGe.setIterProb ( s.second, 5, G(s.first[3].second), a )
      :  mG_BOT.setIterProb ( s.second, a );
    return pr;
  }
};
```

Evaluation as a Performance Model

Implemented using 'Modelblocks' C++ templates for sequence models:

$$\mathrm{P}_{\theta_Y}(s_t^{1..D} \mid s_{t-1}^{1..D}) = \sum_{r_t^{1..D}} \mathrm{P}_{\theta_{\text{Reduce}}}(r_t^{1..D} \mid s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{Shift}}}(s_t^{1..D} \mid r_t^{1..D}\ s_{t-1}^{1..D})$$
$$\stackrel{\text{def}}{=} \sum_{r_t^{1..D}} \prod_{d=1}^{D} \mathrm{P}_{\theta_{R,d}}(r_t^d \mid r_t^{d+1} s_{t-1}^d s_{t-1}^{d-1}) \cdot \mathrm{P}_{\theta_{S,d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t-1}^d s_t^{d-1})$$

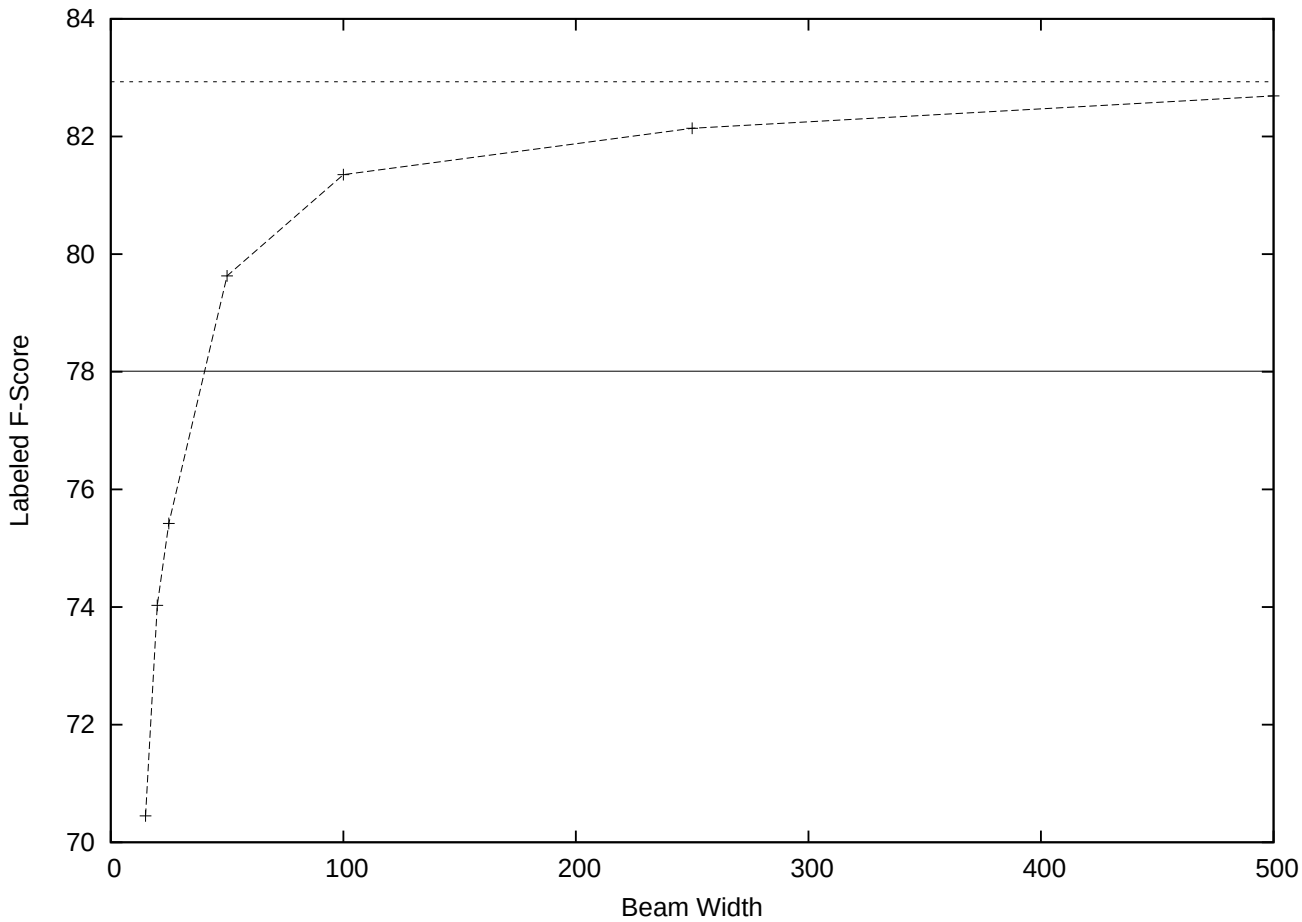
Factored model is compiled into HMM Viterbi recognizer using template:

```
#include "TextObsVars.h"
#include "HHMMLangModel-gf.h"
#include "TextObsModel.h"
#include "HHMMParser.h"

int main (int nArgs, char* argv[]) {
  HMM_Viterbi_MLS<YModel,XModel,S,R> ( nArgs, argv );

}
```

Optimized with 'lazy' k-best search across variables at each time step.

Evaluation as a Performance Model

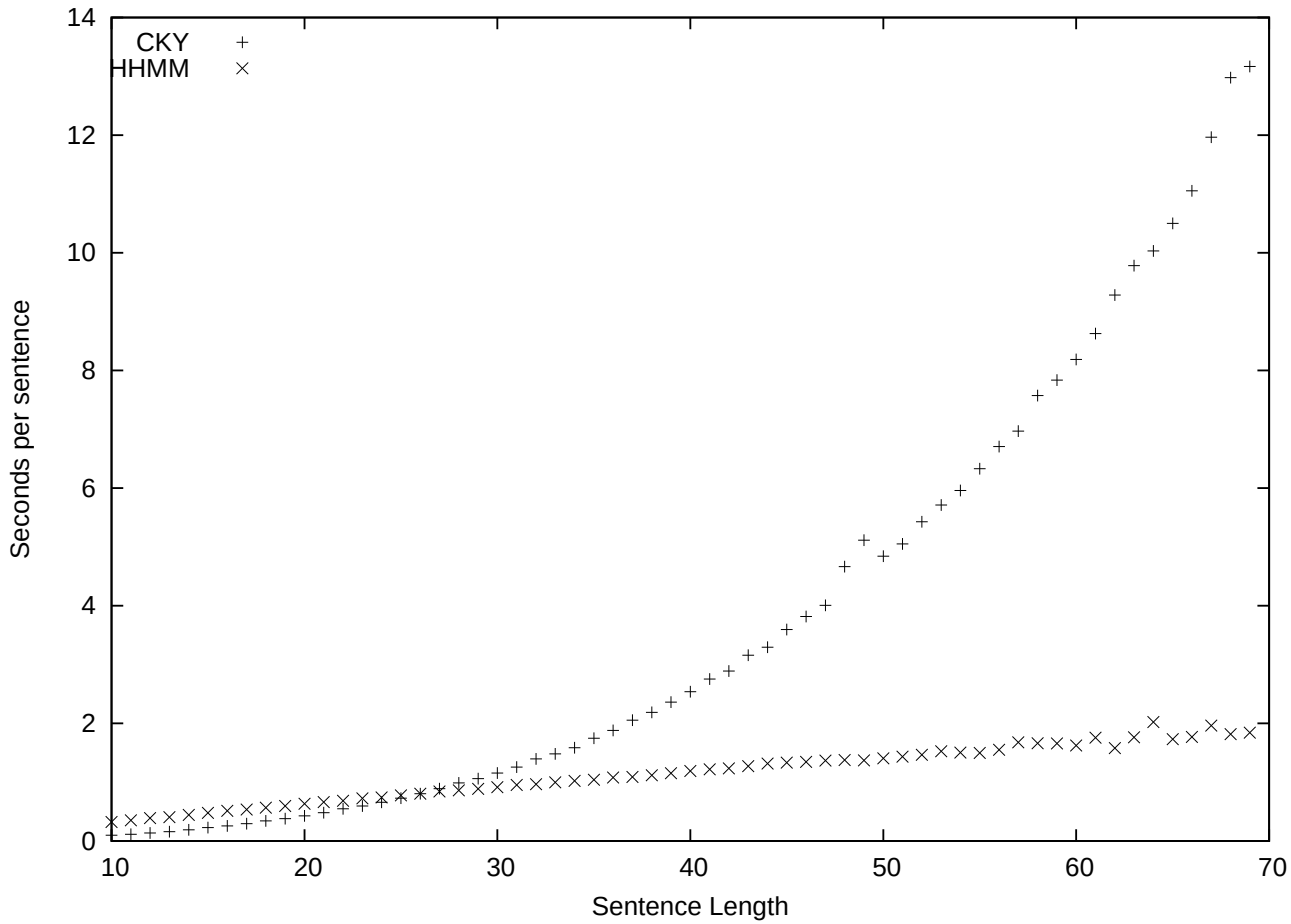
Parser still accurate at narrow beam widths [Miller and Schuler, 2010]:



Accuracy on WSJ sect 23 using beam width 15, 20, 25, 50, 100, 250, 500.

Evaluation as a Performance Model

Speed competitive with CKY parsing [Miller and Schuler, 2010]:



HHMM beam 20 (74%F) vs. 'vanilla' CKY [Klein & Manning,'03] (71%F).

Evaluation as a Performance Model

Does optionally arc-eager strategy support surprisal on reading time? [Hale, 2001, Roark et al., 2009]

Evaluation as a Performance Model

Does optionally arc-eager strategy support surprisal on reading time? [Hale, 2001, Roark et al., 2009]

Evaluated correlation with reading times on [Bachrach et al., 2009]:
"joe was a big bear of a man six feet six inches tall and barrel-chested"
"when he fell off the moon tower the tremor his fall caused seemed not unlike an earthquake to people who lived by"

Evaluation as a Performance Model

Does optionally arc-eager strategy support surprisal on reading time? [Hale, 2001, Roark et al., 2009]

Evaluated correlation with reading times on [Bachrach et al., 2009]:
"joe was a big bear of a man six feet six inches tall and barrel-chested"
"when he fell off the moon tower the tremor his fall caused seemed not unlike an earthquake to people who lived by"

Factors:

Factor	Description	Expected
word order	read faster as story goes on	neg. slope
reciprocal length	read longer words slower	pos. slope
unigram freq.	read common words faster	neg. slope
bigram prob.	read common bigrams faster	neg. slope
embedding diff.	read embedded phrases slower	pos. slope
entropy reduction	read perplexing words slower	pos. slope
surprisal	read surprising words slower	pos. slope

Evaluation as a Performance Model

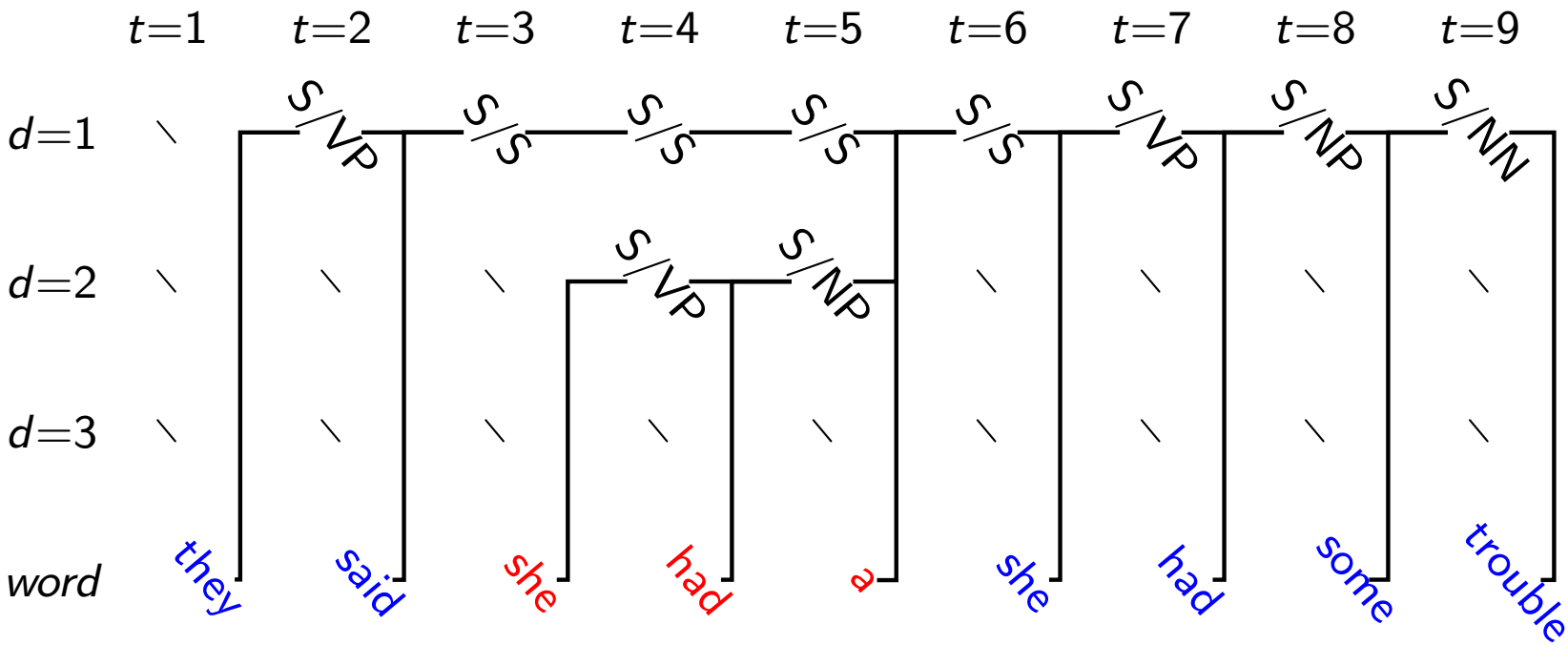
Yes! Results of linear mixed-effect modeling [Wu et al., 2010]:

Factor	Coefficient	Std. Err.	t-value
(Intercept)	$-9.340 \cdot 10^{-3}$	$5.347 \cdot 10^{-2}$	-0.175
word order	$-3.746 \cdot 10^{-5}$	$7.808 \cdot 10^{-6}$	-4.797*
reciprocal length	$-2.002 \cdot 10^{-2}$	$1.635 \cdot 10^{-2}$	-1.225
unigram freq.	$-8.090 \cdot 10^{-2}$	$3.690 \cdot 10^{-1}$	-0.219
bigram prob.	$-2.074 \cdot 10^{+0}$	$8.132 \cdot 10^{-1}$	-2.551*
embedding diff.	$9.390 \cdot 10^{-3}$	$3.268 \cdot 10^{-3}$	2.873*
entropy reduction	$2.753 \cdot 10^{-2}$	$6.792 \cdot 10^{-3}$	4.052*
surprisal	$3.950 \cdot 10^{-3}$	$3.452 \cdot 10^{-4}$	11.442*

Significance (indicated by *) is reported at $p < 0.05$.

Evaluation as a Performance Model

Incomplete constituents also provide nice account of disfluency [Miller and Schuler, 2008, Miller, 2009]



'She had a' is reparandum; intended to be replaced w. 'she had some trouble.'

Here, incomplete constituent can be fluent ($\approx$ conjunction) until repair.

Overview

Tutorial talk:

- Part I: Incremental Parsing
  - bounded-memory sequence model
  - connection to phrase structure
  - coverage
  - implementation/evaluation as performance model
- Part II: Extensions (Semantic Dependencies)
  - preserving probabilistic dependencies in sequence model
  - preserving semantic dependencies in sequence model
  - interactive speech interpretation
  - an analysis of non-local dependencies

Preserving Dependencies in Sequence Model

Transformation on trees is ok for syntax,
but to study effects of lexicalization / selectional restrictions / semantics,
need to preserve dependencies...

Preserving Dependencies in Sequence Model

Transformation on trees is ok for syntax,
but to study effects of lexicalization / selectional restrictions / semantics,
need to preserve dependencies...

Define factored HMM probabilities in terms of original CFG:

1) Start by decomposing yield into yields of left and right children

$$\bar{x}_\eta = \bar{x}_{\eta \cdot 0} \cdot \bar{x}_{\eta \cdot 1}$$

2) Equivalently, decompose into yield of incomplete & complete constituent

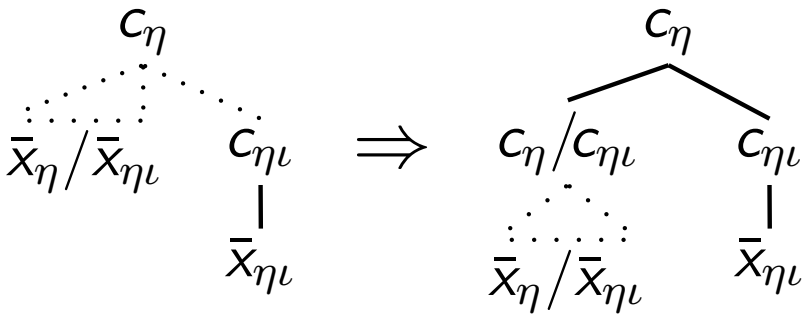
$$\bar{x}_\eta = (\bar{x}_\eta / \bar{x}_{\eta \cdot \iota}) \cdot \bar{x}_{\eta \cdot \iota} \quad \text{for all } \iota \in 1^+$$

Preserving Dependencies in Sequence Model

3) Using these yields, define recurrences analogous to transform rules:

$$\mathrm{P}_{\theta_{\mathrm{Ins(G)}}}(\bar{x}_\eta \mid c_\eta) = \sum_{\iota \in 1^+, \bar{x}_{\eta\iota}, c_{\eta\iota}} \mathrm{P}_{\theta_{\mathrm{IC(G)}}}(\bar{x}_\eta / \bar{x}_{\eta\iota},\ c_{\eta\iota} \mid c_\eta) \cdot \mathrm{P}_{\theta_{\mathrm{Ins(G)}}}(\bar{x}_{\eta\iota} \mid c_{\eta\iota})$$

analogous to 'end case':
(forward application)



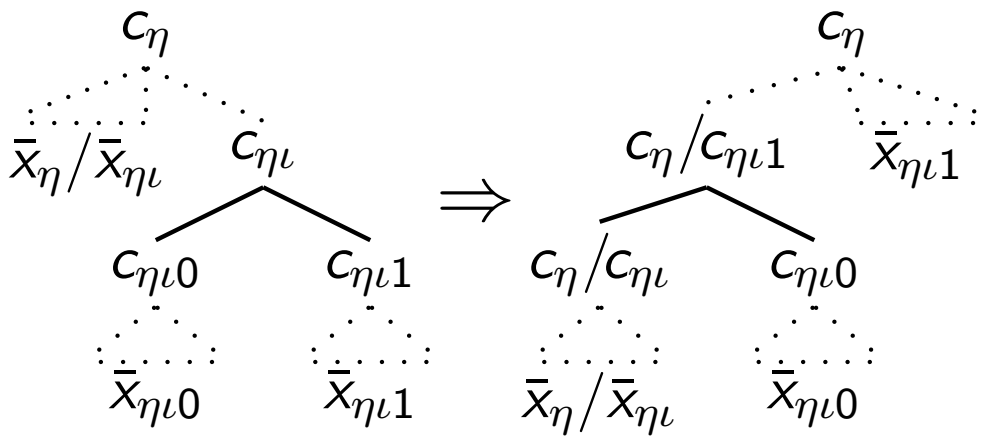
Preserving Dependencies in Sequence Model

3) Using these yields, define recurrences analogous to transform rules:

$$\mathrm{P}_{\theta_{\mathrm{IC(G)}}}(\bar{x}_\eta/\bar{x}_{\eta\iota 1},\ c_{\eta\iota 1} \mid c_\eta) =$$

$$\sum_{\bar{x}_{\eta\iota}, c_{\eta\iota}} \mathrm{P}_{\theta_{\mathrm{IC(G)}}}(\bar{x}_\eta/\bar{x}_{\eta\iota},\ c_{\eta\iota} \mid c_\eta) \cdot \sum_{\bar{x}_{\eta\iota 0}, c_{\eta\iota 0}} \mathrm{P}_{\theta_{\mathrm{G}}}(c_{\eta\iota} \rightarrow c_{\eta\iota 0}\ c_{\eta\iota 1}) \cdot \mathrm{P}_{\theta_{\mathrm{Ins(G)}}}(\bar{x}_{\eta\iota 0} \mid c_{\eta\iota 0})$$

analogous to 'middle case':
(forward composition)

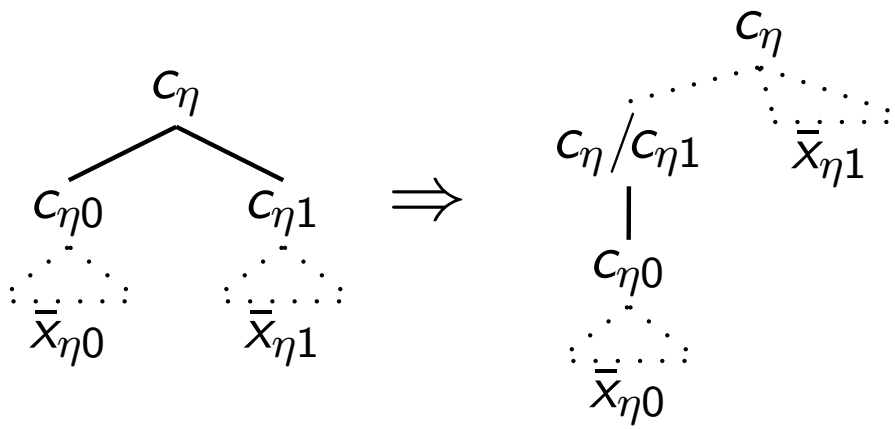


Preserving Dependencies in Sequence Model

3) Using these yields, define recurrences analogous to transform rules:

$$\mathrm{P}_{\theta_{\mathrm{IC(G)}}}(\bar{x}_\eta/\bar{x}_{\eta 1},\ c_{\eta 1} \mid c_\eta) = \sum_{\bar{x}_{\eta 0}, c_{\eta 0}} \mathrm{P}_{\theta_{\mathrm{G}}}(c_\eta \rightarrow c_{\eta 0}\ c_{\eta 1}) \cdot \mathrm{P}_{\theta_{\mathrm{Ins(G)}}}(\bar{x}_{\eta 0} \mid c_{\eta 0})$$

analogous to 'begin case':
(type raising)



Preserving Dependencies in Sequence Model

4) Connect each incomplete constituent as leftmost descendant of previous:

$$\mathsf{P}_{\theta_{\mathsf{Fwd}}}((\bar{x}_{\eta_d}/\bar{x}_{\eta_d\iota_d})_{d=1}^{D}, (c_{\eta_d})_{d=1}^{D}, (c_{\eta_d\iota_d})_{d=1}^{D}) =$$

$$\prod_{d=1}^{D} \mathsf{P}_{\theta_{\text{G-RL}^*,d}}(c_{\eta_{d-1}\iota_{d-1}} \xrightarrow{*} c_{\eta_d} \ldots) \cdot \mathsf{P}_{\theta_{\mathsf{IC(G)}}}(\bar{x}_{\eta_d}/\bar{x}_{\eta_d\iota_d}, c_{\eta_d\iota_d} \mid c_{\eta_d})$$

Preserving Dependencies in Sequence Model

4) Connect each incomplete constituent as leftmost descendant of previous:

$$\mathrm{P}_{\theta_{\mathrm{Fwd}}}((\bar{x}_{\eta_d}/\bar{x}_{\eta_d\iota_d})_{d=1}^{D},(c_{\eta_d})_{d=1}^{D},(c_{\eta_d\iota_d})_{d=1}^{D}) = \prod_{d=1}^{D} \mathrm{P}_{\theta_{\text{G-RL}^*,d}}(c_{\eta_{d-1}\iota_{d-1}} \xrightarrow{*} c_{\eta_d}\ldots) \cdot \mathrm{P}_{\theta_{\mathrm{IC(G)}}}(\bar{x}_{\eta_d}/\bar{x}_{\eta_d\iota_d}, c_{\eta_d\iota_d} \mid c_{\eta_d})$$

with leftmost-descendant terms defined by value iteration [Bellman, 1957]:

$$\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{0} c_{\eta 0}\ldots) = \sum_{c_{\eta 1}} \mathrm{P}_{\theta_{\text{G-R},d}}(c_\eta \rightarrow c_{\eta 0}\ c_{\eta 1})$$

$$\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{k} c_{\eta 0^k 0}\ldots) = \sum_{c_{\eta 0^k}, c_{\eta 0^k 1}} \mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{k-1} c_{\eta 0^k}\ldots) \cdot \mathrm{P}_{\theta_{\text{G-L},d}}(c_{\eta 0^k} \rightarrow c_{\eta 0^k 0}\ c_{\eta 0^k 1})$$

$$\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{*} c_{\eta\iota}\ldots) = \sum_{k=0}^{\infty} \mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{k} c_{\eta\iota}\ldots)$$

$$\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{+} c_{\eta\iota}\ldots) = \mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{*} c_{\eta\iota}\ldots) - \mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{0} c_{\eta\iota}\ldots)$$

Preserving Dependencies in Sequence Model

5) Then, associating store variables with incomplete constituent categories...

$$s_t^d \stackrel{\text{def}}{=} \langle c_{\eta_d}, c_{\eta_d \iota_d} \rangle \; s.t. \; \iota_d \in 1^+, \eta_d \in \eta_{d-1} \iota_{d-1} 0^+$$

$$r_t^d \stackrel{\text{def}}{=} \langle c_{\eta_d}, f_{r_t^d} \rangle \quad s.t. \; f_{r_t^d} = [\![\bar{x}_{\eta_d} = c_{\eta_d}]\!] \quad \text{(reduction or not)}$$

Preserving Dependencies in Sequence Model

5) Then, associating store variables with incomplete constituent categories...

$$s_t^d \stackrel{\text{def}}{=} \langle c_{\eta_d}, c_{\eta_d \iota_d} \rangle \;\; s.t. \;\; \iota_d \in 1^+, \eta_d \in \eta_{d-1} \iota_{d-1} 0^+$$

$$r_t^d \stackrel{\text{def}}{=} \langle c_{\eta_d}, f_{r_t^d} \rangle \quad s.t. \;\; f_{r_t^d} = [\![\bar{x}_{\eta_d} = c_{\eta_d}]\!] \quad \text{(reduction or not)}$$

divide incomplete constituents into components of HMM recurrence...

$$\begin{aligned} \mathrm{P}_{\theta_{\text{Fwd}}}(x_{1..t}\, s_t^{1..D}) &= \sum_{\eta_{1..D}, \iota_{1..D}} \mathrm{P}_{\theta_{\text{Fwd}}}((\bar{x}_{\eta_d} / \bar{x}_{\eta_d \iota_d})_{d=1}^D, (c_{\eta_d})_{d=1}^D, (c_{\eta_d \iota_d})_{d=1}^D) \\ &= \sum_{s_{t-1}^{1..D}} \mathrm{P}_{\theta_{\text{Fwd}}}(x_{1..t-1}\, s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{Y}}}(s_t^{1..D} \mid s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{X}}}(x_t \mid s_t^{1..D}) \end{aligned}$$

Preserving Dependencies in Sequence Model

5) Then, associating store variables with incomplete constituent categories...

$$s_t^d \stackrel{\text{def}}{=} \langle c_{\eta_d}, c_{\eta_d \iota_d} \rangle \;\; s.t. \;\; \iota_d \in 1^+, \eta_d \in \eta_{d-1} \iota_{d-1} 0^+$$

$$r_t^d \stackrel{\text{def}}{=} \langle c_{\eta_d}, f_{r_t^d} \rangle \quad s.t. \;\; f_{r_t^d} = [\![\bar{x}_{\eta_d} = c_{\eta_d}]\!] \quad \text{(reduction or not)}$$

divide incomplete constituents into components of HMM recurrence...

$$\begin{aligned} \mathrm{P}_{\theta_{\text{Fwd}}}(x_{1..t}\, s_t^{1..D}) &= \sum_{\eta_{1..D}, \iota_{1..D}} \mathrm{P}_{\theta_{\text{Fwd}}}((\bar{x}_{\eta_d} / \bar{x}_{\eta_d \iota_d})_{d=1}^D, (c_{\eta_d})_{d=1}^D, (c_{\eta_d \iota_d})_{d=1}^D) \\ &= \sum_{s_{t-1}^{1..D}} \mathrm{P}_{\theta_{\text{Fwd}}}(x_{1..t-1}\, s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{Y}}}(s_t^{1..D} \mid s_{t-1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{X}}}(x_t \mid s_t^{1..D}) \end{aligned}$$

using the hidden state model already defined:

$$\begin{aligned} \mathrm{P}_{\theta_{\text{Y}}}(s_t^{1..D} \mid s_{t\text{-}1}^{1..D}) &= \sum_{r_t^{1..D}} \mathrm{P}_{\theta_{\text{Reduce}}}(r_t^{1..D} \mid s_{t\text{-}1}^{1..D}) \cdot \mathrm{P}_{\theta_{\text{Shift}}}(s_t^{1..D} \mid r_t^{1..D}\; s_{t\text{-}1}^{1..D}) \\ &\stackrel{\text{def}}{=} \sum_{r_t^{1..D}} \prod_{d=1}^{D} \mathrm{P}_{\theta_{\text{R},d}}(r_t^d \mid r_t^{d+1} s_{t\text{-}1}^d s_{t\text{-}1}^{d\text{-}1}) \cdot \mathrm{P}_{\theta_{\text{S},d}}(s_t^d \mid r_t^{d+1} r_t^d\; s_{t\text{-}1}^d s_t^{d\text{-}1}) \end{aligned}$$

Preserving Dependencies in Sequence Model

Model uses reduction states r_t^d to ensure only one transition per time step (all reductions below some depth, none above; transition is at crossover)

$$\mathrm{P}_{\theta_{\mathrm{R},d}}(r_t^d \mid r_t^{d+1} s_{t-1}^d s_{t-1}^{d-1}) \stackrel{\text{def}}{=} \begin{cases} \text{if } f_{r_t^{d+1}} = 0 : [\![r_t^d = \mathbf{r}_\perp]\!] \\ \text{if } f_{r_t^{d+1}} = 1 : \mathrm{P}_{\theta_{\text{R-R},d}}(r_t^d \mid r_t^{d+1}\ s_{t-1}^d\ s_{t-1}^{d-1}) \end{cases}$$

$$\mathrm{P}_{\theta_{\mathrm{S},d}}(s_t^d \mid r_t^{d+1} r_t^d\ \ s_{t-1}^d s_t^{d-1}) \stackrel{\text{def}}{=} \begin{cases} \text{if } f_{r_t^{d+1}} = 1,\ f_{r_t^d} = 1 : \mathrm{P}_{\theta_{\text{S-E},d}}(s_t^d \mid s_t^{d-1}) \\ \text{if } f_{r_t^{d+1}} = 1,\ f_{r_t^d} = 0 : \mathrm{P}_{\theta_{\text{S-T},d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t-1}^d s_t^{d-1}) \\ \text{if } f_{r_t^{d+1}} = 0,\ f_{r_t^d} = 0 : [\![s_t^d = s_{t-1}^d]\!] \end{cases}$$

Models now defined on CFG probs + chopped-up left-descendant terms...

Expansion probabilities simply contribute new left-descendant term:

$$\mathrm{P}_{\theta_{\text{S-E},d}}(\langle c_{\eta\iota}, c'_{\eta\iota} \rangle \mid \langle -, c_\eta \rangle) \stackrel{\text{def}}{=} \mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \stackrel{*}{\rightarrow} c_{\eta\iota}\ \ldots) \cdot [\![x_{\eta\iota} = c'_{\eta\iota} = c_{\eta\iota}]\!]$$

Preserving Dependencies in Sequence Model

Models now defined on CFG probs + chopped-up left-descendant terms...

Reductions divide left-descendant prob into $0/{>}0$ left-expansions:

$$\mathsf{P}_{\theta_{\text{R-R},d}}(r_t^d \mid r_t^{d+1} s_{t-1}^d s_{t-1}^{d-1}) \stackrel{\text{def}}{=} \begin{cases} \text{if } c_{r_t^{d+1}} \neq x_t : [\![r_t^d = \mathbf{r}_\perp]\!] \\ \text{if } c_{r_t^{d+1}} = x_t : \mathsf{P}_{\theta_{\text{R-R},d}}(r_t^d \mid s_{t-1}^d s_{t-1}^{d-1}) \end{cases}$$

$$\mathsf{P}_{\theta_{\text{R-R},d}}(c_{\eta\iota}, \mathbf{1} \mid \langle -, c_\eta \rangle \ \langle c'_{\eta\iota}, - \rangle) \stackrel{\text{def}}{=} [\![c_{\eta\iota} = c'_{\eta\iota}]\!] \cdot \frac{\mathsf{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{0} c_{\eta\iota} \ \ldots)}{\mathsf{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{*} c_{\eta\iota} \ \ldots)}$$

$$\mathsf{P}_{\theta_{\text{R-R},d}}(c_{\eta\iota}, \mathbf{0} \mid \langle -, c_\eta \rangle \ \langle c'_{\eta\iota}, - \rangle) \stackrel{\text{def}}{=} [\![c_{\eta\iota} = c'_{\eta\iota}]\!] \cdot \frac{\mathsf{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{+} c_{\eta\iota} \ \ldots)}{\mathsf{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{*} c_{\eta\iota} \ \ldots)}$$

Preserving Dependencies in Sequence Model

Models now defined on CFG probs + chopped-up left-descendant terms...

Transitions apply left/right CFG rules above/below incomplete constituent:

$$\mathrm{P}_{\theta_{\text{S-T},d}}(s_t^d \mid r_t^{d+1}\, r_t^d\, s_{t-1}^d\, s_t^{d-1}) \stackrel{\text{def}}{=} \begin{cases} \text{if } r_t^d \neq \mathbf{r}_\perp : \mathrm{P}_{\theta_{\text{S-T-A},d}}(s_t^d \mid s_t^{d-1}\, r_t^d) \\ \text{if } r_t^d = \mathbf{r}_\perp : \mathrm{P}_{\theta_{\text{S-T-W},d}}(s_t^d \mid s_{t-1}^d\, r_t^{d+1}) \end{cases}$$

$$\mathrm{P}_{\theta_{\text{S-T-A},d}}(\langle c_{\eta\iota}, c_{\eta\iota 1}\rangle \mid \langle -, c_\eta\rangle\ c_{\eta\iota 0}) \stackrel{\text{def}}{=} \frac{\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{*} c_{\eta\iota}\ \ldots) \cdot \mathrm{P}_{\theta_{\text{G-L},d}}(c_{\eta\iota} \rightarrow c_{\eta\iota 0}\ c_{\eta\iota 1})}{\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{+} c_{\eta\iota 0}\ \ldots)}$$

$$\mathrm{P}_{\theta_{\text{S-T-W},d}}(\langle c_\eta, c_{\eta\iota 1}\rangle \mid \langle c'_\eta, c_{\eta\iota}\rangle\ c_{\eta\iota 0}) \stackrel{\text{def}}{=} [\![c_\eta = c'_\eta]\!] \cdot \frac{\mathrm{P}_{\theta_{\text{G-R},d}}(c_{\eta\iota} \rightarrow c_{\eta\iota 0}\ c_{\eta\iota 1})}{\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_{\eta\iota} \xrightarrow{0} c_{\eta\iota 0}\ \ldots)}$$

Preserving Dependencies in Sequence Model

Models now defined on CFG probs + chopped-up left-descendant terms...

Transitions apply left/right CFG rules above/below incomplete constituent:

$$\mathrm{P}_{\theta_{\text{S-T},d}}(s_t^d \mid r_t^{d+1}\, r_t^d\; s_{t-1}^d\, s_t^{d-1}) \stackrel{\text{def}}{=} \begin{cases} \text{if } r_t^d \neq \mathbf{r}_\perp : \mathrm{P}_{\theta_{\text{S-T-A},d}}(s_t^d \mid s_t^{d-1}\, r_t^d) \\ \text{if } r_t^d = \mathbf{r}_\perp : \mathrm{P}_{\theta_{\text{S-T-W},d}}(s_t^d \mid s_{t-1}^d\, r_t^{d+1}) \end{cases}$$

$$\mathrm{P}_{\theta_{\text{S-T-A},d}}(\langle c_{\eta\iota}, c_{\eta\iota 1}\rangle \mid \langle -, c_\eta\rangle\; c_{\eta\iota 0}) \stackrel{\text{def}}{=} \frac{\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{*} c_{\eta\iota}\;\ldots) \cdot \mathrm{P}_{\theta_{\text{G-L},d}}(c_{\eta\iota} \rightarrow c_{\eta\iota 0}\;\; c_{\eta\iota 1})}{\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_\eta \xrightarrow{+} c_{\eta\iota 0}\;\ldots)}$$

$$\mathrm{P}_{\theta_{\text{S-T-W},d}}(\langle c_\eta, c_{\eta\iota 1}\rangle \mid \langle c'_\eta, c_{\eta\iota}\rangle\; c_{\eta\iota 0}) \stackrel{\text{def}}{=} [\![c_\eta = c'_\eta]\!] \cdot \frac{\mathrm{P}_{\theta_{\text{G-R},d}}(c_{\eta\iota} \rightarrow c_{\eta\iota 0}\;\; c_{\eta\iota 1})}{\mathrm{E}_{\theta_{\text{G-RL}^*,d}}(c_{\eta\iota} \xrightarrow{0} c_{\eta\iota 0}\;\ldots)}$$

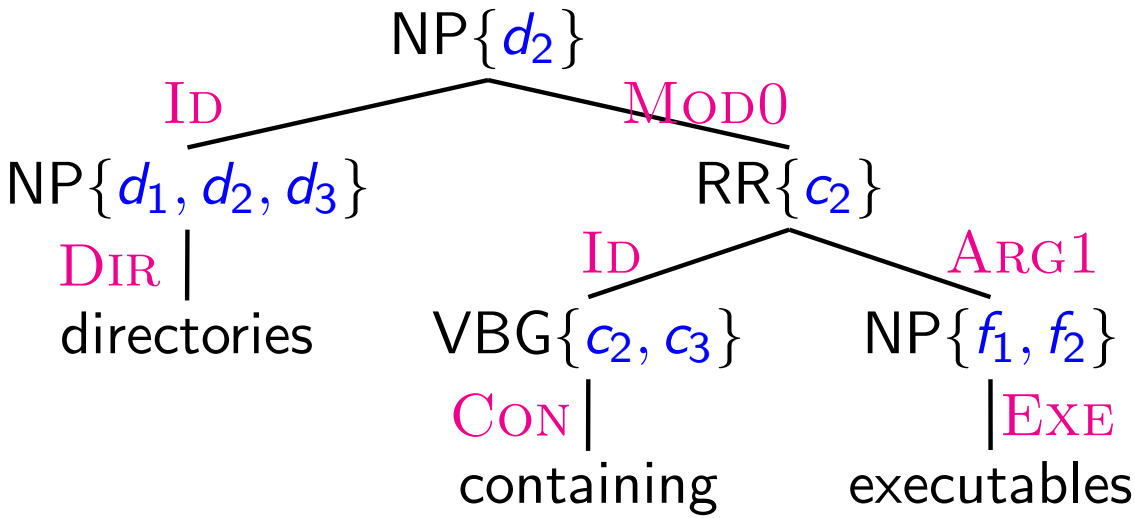
Model thus preserves conditional dependencies of original PCFG.

Same probabilities as CKY (mod. depth bounds): incrementalize any PCFG!

Right-Corner Transform on Operator Chains

Preserving CFG dependencies preserves many predicate-argument relations.

Allows incremental interpreter def. on bottom-up (compositional) semantics:



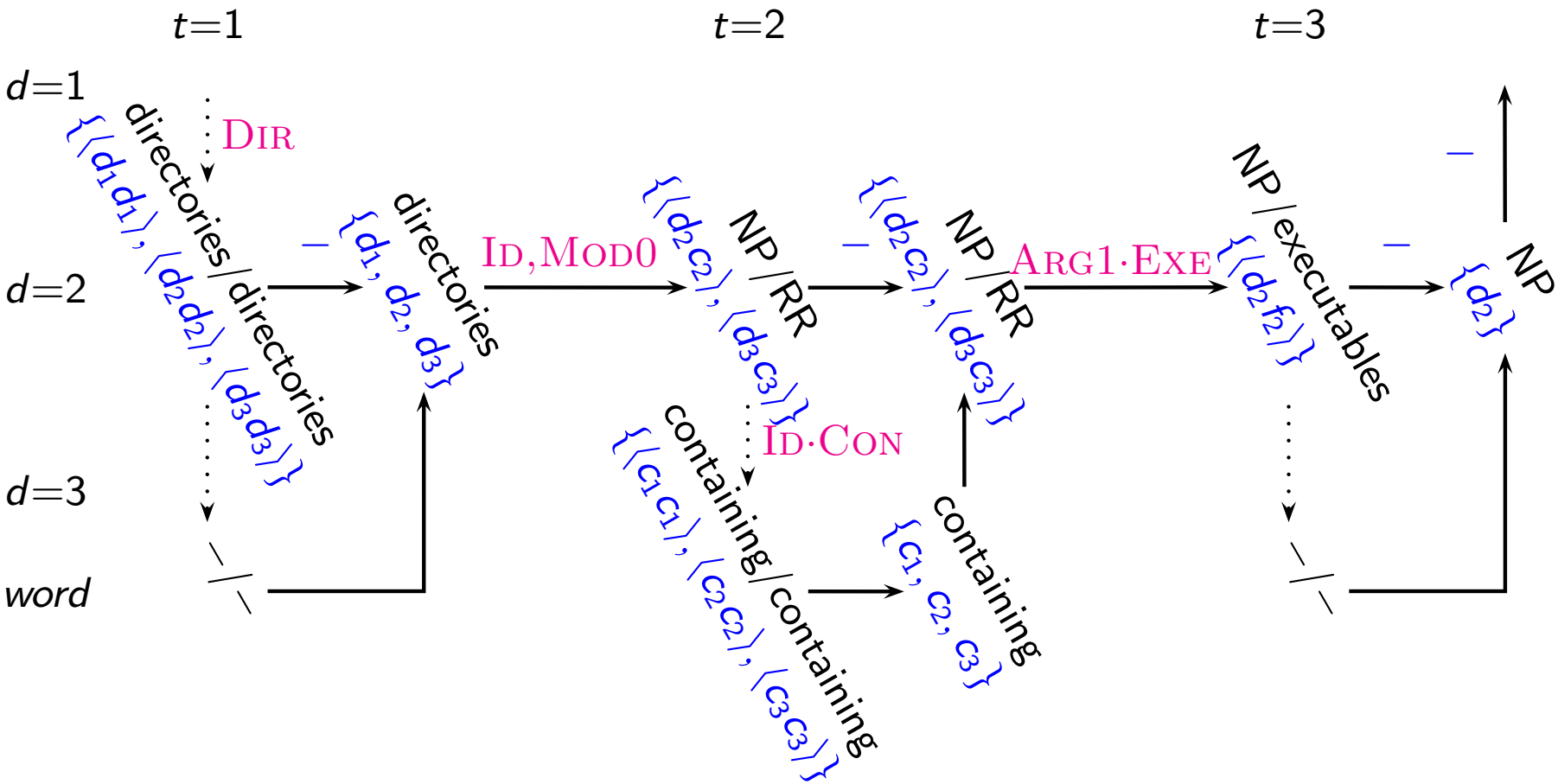
For example, interpretation may use probabilistic dependencies explicitly:

$$\mathrm{P}_{\theta_\mathrm{G}}(lci_\eta \rightarrow lci_{\eta 0}\ lci_{\eta 1}) = \mathrm{P}_{\theta_\mathrm{M}}(lc_{\eta 0}, lc_{\eta 1} \mid lci_\eta) \cdot \mathrm{P}_{\theta_\mathrm{L}}(i_{\eta 0} \mid l_{\eta 0}, i_\eta) \cdot \mathrm{P}_{\theta_\mathrm{L}}(i_{\eta 1} \mid l_{\eta 1}, i_\eta)$$

(referent i_η generates child relns, cats $lc_{\eta 0/1}$; relns generate referents $i_{\eta 0/1}$)

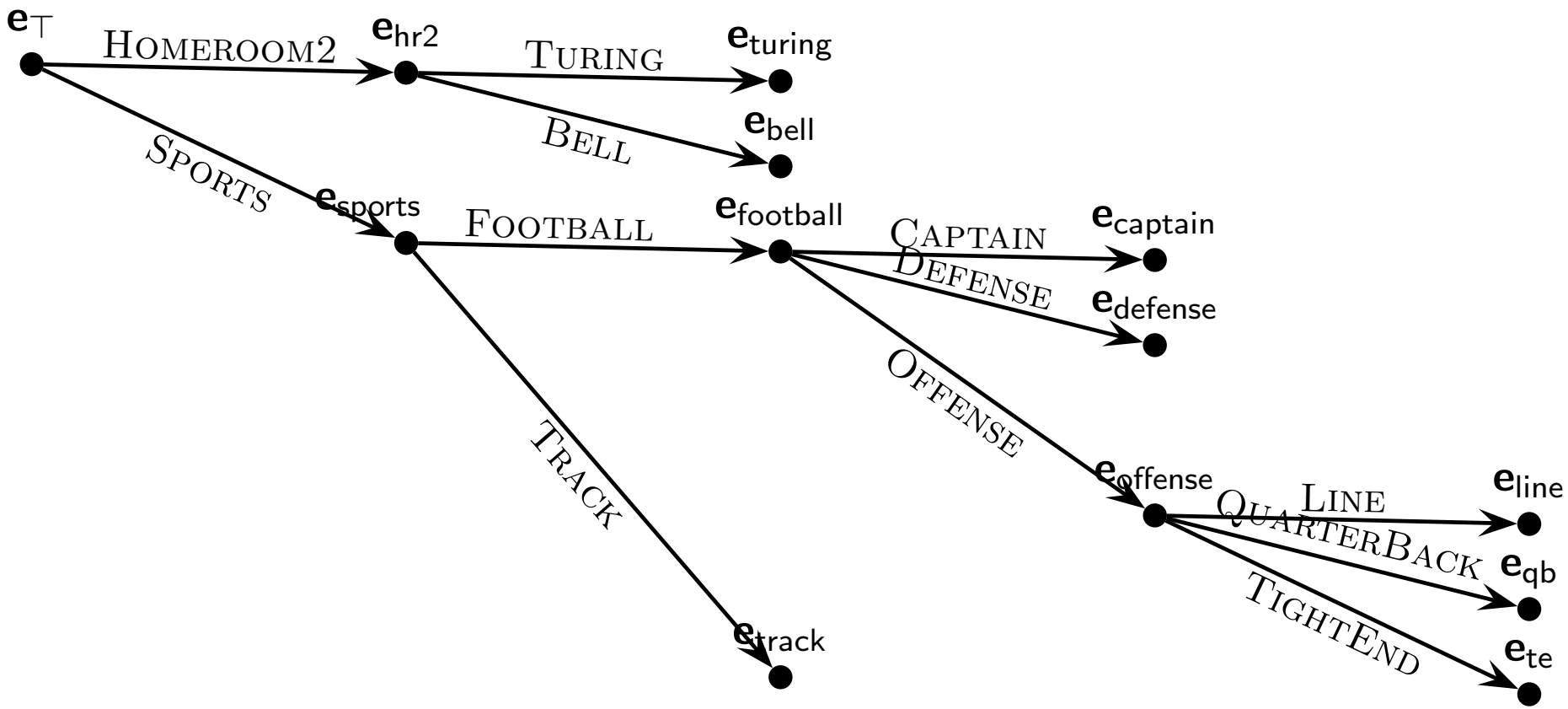
Right-Corner Transform on Operator Chains

Transformed operations aligned to sequence model:



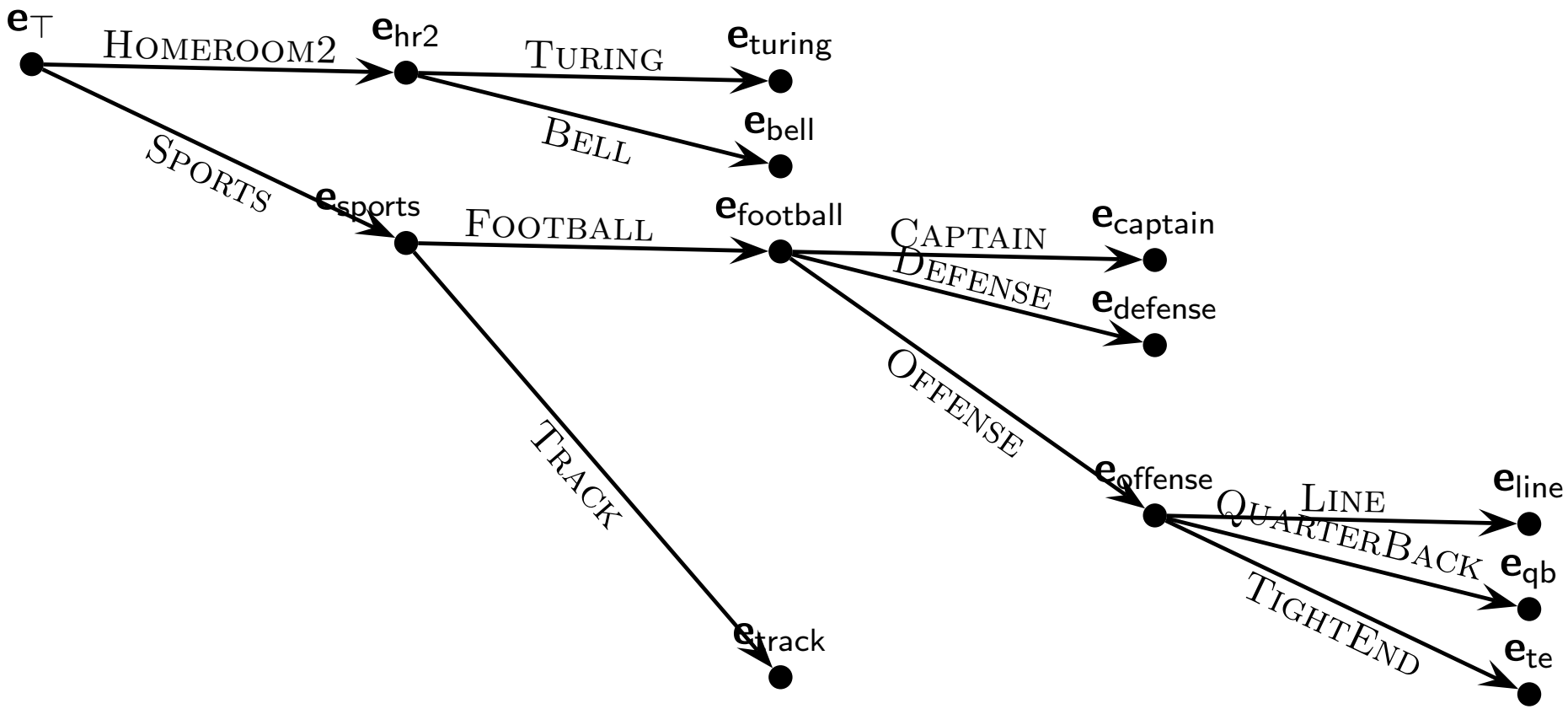
Evaluation: first-order denotations

Real-time speech interface (acoust model: Robinson'94, domain: 240 indiv.)
Student domain: 'go to sports track and set Homeroom 2 Bell to captain'



Evaluation: first-order denotations

Real-time speech interface (acoust model: Robinson'94, domain: 240 indiv.)
Student domain: 'go to sports track and set Homeroom 2 Bell to captain'

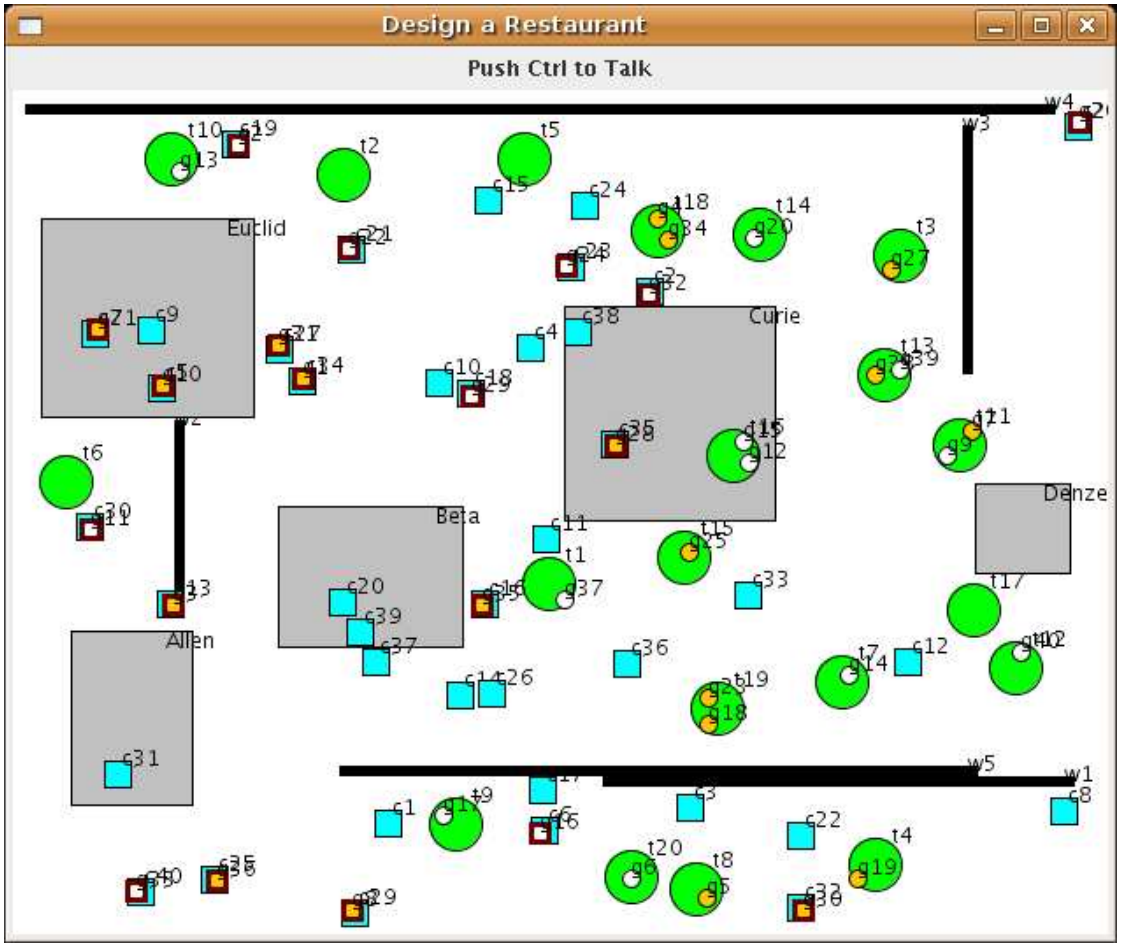


Interactive model closer to human than trigram: concept error 26% → 17%

Evaluation: second-order denotations

Bounded model allows 2nd order denotations in real time speech (13% SER)

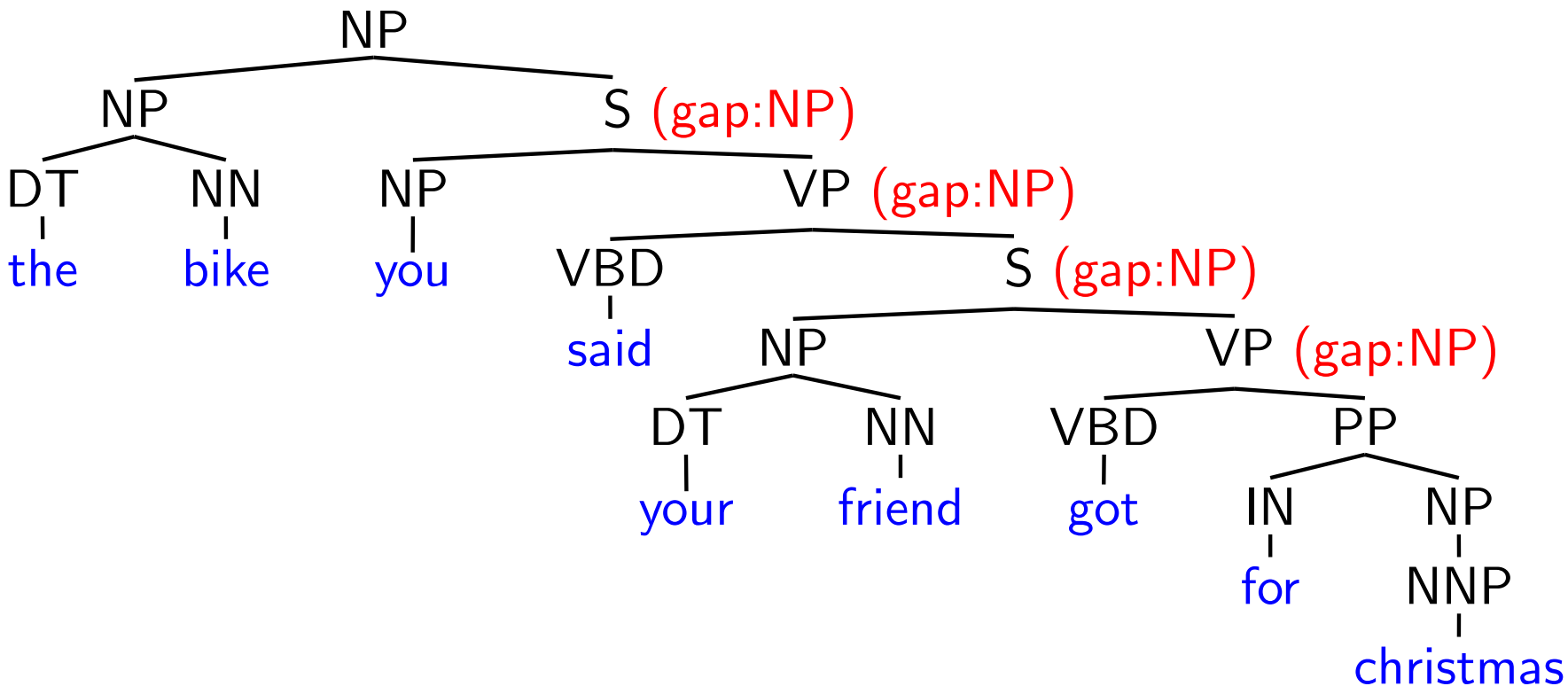
Restaurant domain: 'select the glasses on chairs'



Non-local Dependencies

Model can also introduce non-CFG dependencies across store elements:

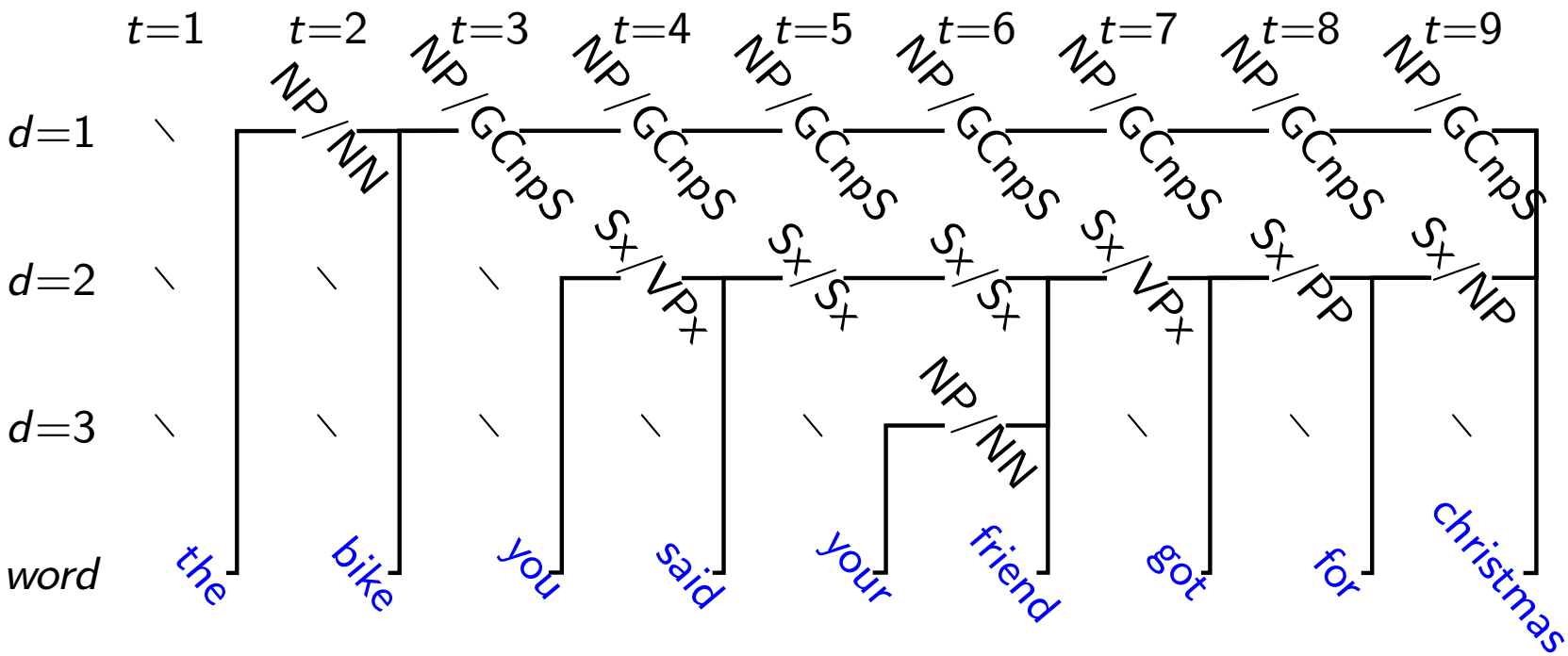
Consider filler-gap construction:



Conventional analysis requires feature passing [Pollard and Sag, 1994].

Non-local Dependencies

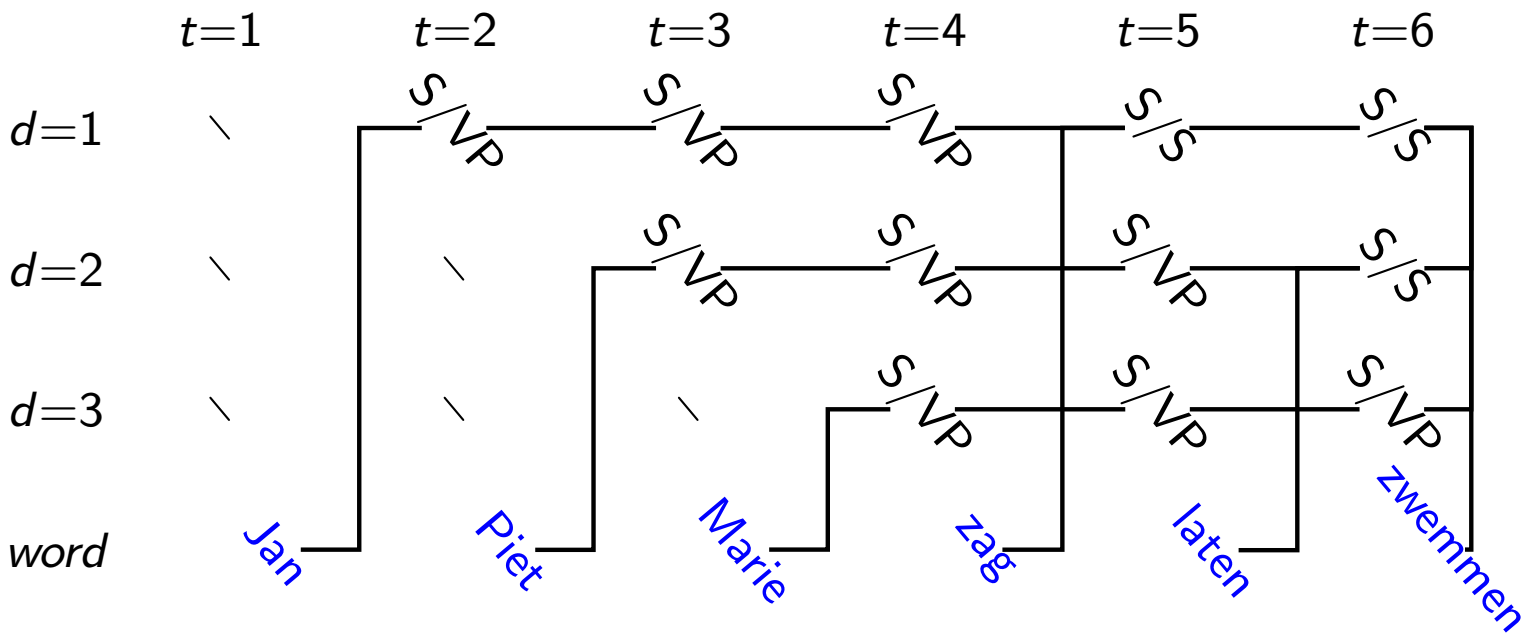
In sequence model, referent is locally available at gap position:



Still need 'x' feature to require gap, but no need to pass referent/category (obtained from tree by restricting awaited transitions on GC category)

Non-local Dependencies

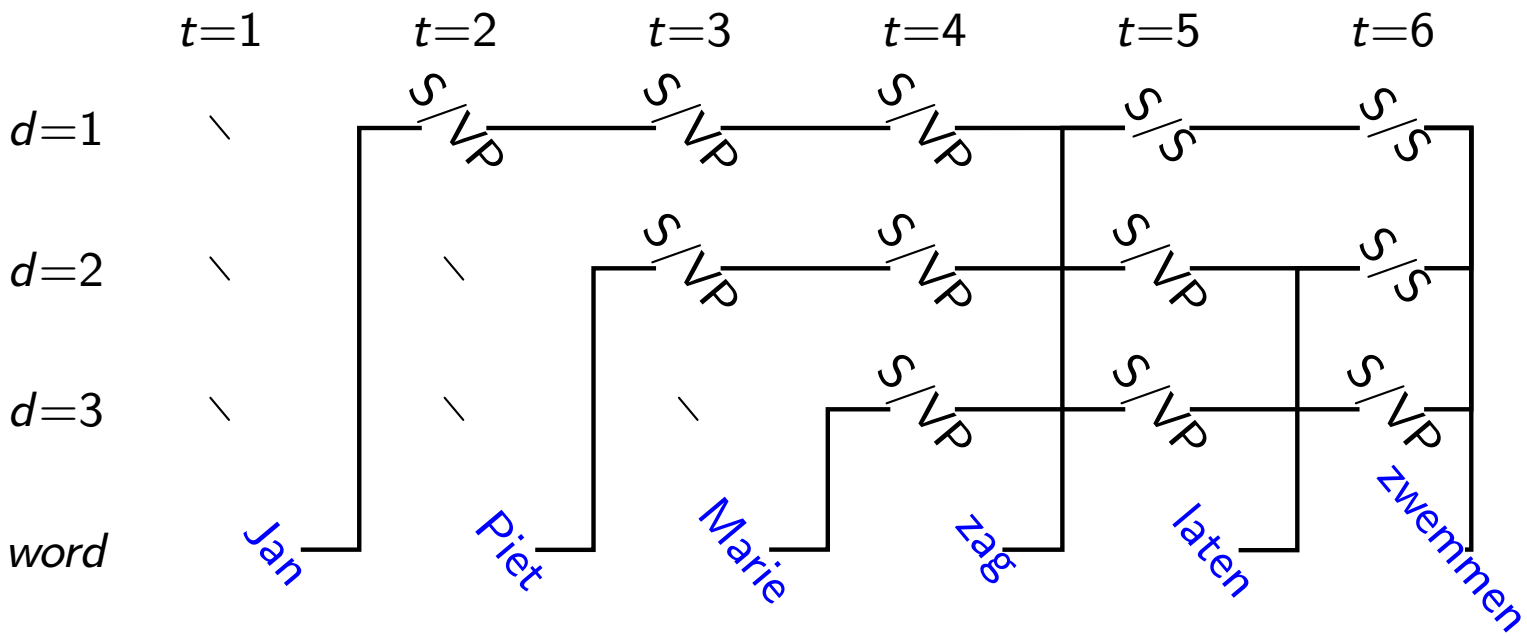
Crossed and nested dependencies respect chunks,
but require some relaxation in definition of reduction:



(no longer easily associated with phrase structure tree)

Non-local Dependencies

Crossed and nested dependencies respect chunks,
but require some relaxation in definition of reduction:



(no longer easily associated with phrase structure tree)

Speculate EPDA-style transition operations could be learned,
but PDA-style (HHMM) operations usually minimize surprisal.

Conclusion

Simple processing model, matches observations about human memory

1. bounded number of unconnected chunks
   [Miller, 1956, Cowan, 2001]
   (subjects group stimuli into only 4 or so clusters)
2. process rich syntax
   [Chomsky and Miller, 1963]
   (center embedding: 'if [neither [the man [the cop] saw] nor ...] then ...')
   (cf. center recursion: '?? the malt [the rat [the cat chased] ate] ...')
3. processing is incremental
   [Sachs, 1967, Jarvella, 1971]
   (subjects can't remember specific words earlier in sentence)
4. processing is parallel, probabilistic
   [Jurafsky, 1996, Hale, 2001, Levy, 2008]
   (probabilistic / info. theoretic measures correlate w. reading times)

Conclusion

Simple processing model, matches observations about human memory

In particular, the factored sequence model (dynamic Bayes net):

1. has easily interpretable random variables
   (explicit estimation of speaker intent; cf. neural net)
2. has clear role of bounded working memory store
   (random variable for each store element)
3. has clear role of syntax
   (grammar transform turns trees into chunks for store elements)
4. is fast enough to interact with real-time speech recognizer
   (using cool engineering tricks: best-first / 'lazy' k-best search)

Result is a nice platform for linguistic experimentation!

Bibliography I


Abney, S. P. and Johnson, M. (1991).
Memory requirements and local ambiguities of parsing strategies.
J. Psycholinguistic Research, 20(3):233–250.


Bachrach, A., Roark, B., Marantz, A., Whitfield-Gabrieli, S., Cardenas, C., and Gabrieli, J. D. (2009).
Incremental prediction in naturalistic language processing: An fMRI study.


Bellman, R. (1957).
Dynamic Programming.
Princeton University Press, Princeton, NJ.


Chomsky, N. and Miller, G. A. (1963).
Introduction to the formal analysis of natural languages.
In *Handbook of Mathematical Psychology*, pages 269–321. Wiley.

Bibliography II


Cowan, N. (2001).
The magical number 4 in short-term memory: A reconsideration of mental storage capacity.
Behavioral and Brain Sciences, 24:87–185.


Hale, J. (2001).
A probabilistic earley parser as a psycholinguistic model.
In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 159–166, Pittsburgh, PA.


Jarvella, R. J. (1971).
Syntactic processing of connected speech.
Journal of Verbal Learning and Verbal Behavior, 10:409–416.

Bibliography III


Johnson, M. (1998).
Finite state approximation of constraint-based grammars using left-corner grammar transforms.
In *Proceedings of COLING/ACL*, pages 619–623, Montreal, Canada.


Jurafsky, D. (1996).
A probabilistic model of lexical and syntactic access and disambiguation.
Cognitive Science: A Multidisciplinary Journal, 20(2):137–194.


Klein, D. and Manning, C. D. (2003).
Accurate unlexicalized parsing.
In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Bibliography IV


Levy, R. (2008).
Expectation-based syntactic comprehension.
Cognition, 106(3):1126–1177.


Miller, G. A. (1956).
The magical number seven, plus or minus two: Some limits on our capacity for processing information.
Psychological Review, 63:81–97.


Miller, T. (2009).
Improved syntactic models for parsing speech with repairs.
In *Proceedings of the North American Association for Computational Linguistics*, Boulder, CO.

Bibliography V


Miller, T. and Schuler, W. (2008).
A syntactic time-series model for parsing fluent and disfluent speech.
In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*.


Miller, T. and Schuler, W. (2010).
Hhmm parsing with limited parallelism.
In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics '10)*.


Pollard, C. and Sag, I. (1994).
Head-driven Phrase Structure Grammar.
University of Chicago Press, Chicago.

Bibliography VI


Roark, B., Bachrach, A., Cardenas, C., and Pallier, C. (2009).
Deriving lexical and syntactic expectation-based measures for psycholinguistic modeling via incremental top-down parsing.
Proceedings of the 2009 Conference on Empirical Methods in Natural Langauge Processing, pages 324–333.


Sachs, J. (1967).
Recognition memory for syntactic and semantic aspects of connected discourse.
Perception and Psychophysics, 2:437–442.


Schuler, W., AbdelRahman, S., Miller, T., and Schwartz, L. (2008).
Toward a psycholinguistically-motivated model of language.
In *Proceedings of COLING*, pages 785–792, Manchester, UK.

Bibliography VII


Schuler, W., AbdelRahman, S., Miller, T., and Schwartz, L. (2010).
Broad-coverage incremental parsing using human-like memory constraints.
Computational Linguistics, 36(1).


Steedman, M. (2000).
The syntactic process.
MIT Press/Bradford Books, Cambridge, MA.


Wu, S., Bachrach, A., Cardenas, C., and Schuler, W. (2010).
Complexity metrics in an incremental right-corner parser.
In *Proceedings of the 49th Annual Conference of the Association for Computational Linguistics (ACL'10)*.