Upcoming deadlines

- Tonight: draft writeup for term project is due
- Tuesday 3/24: the data for your term project must be collected
- Thursday 3/26: HW4 due

Decoding and Inference with Syntactic Translation Models

Machine Translation
Lecture 15

Instructor: Chris Callison-Burch
TAs: Mitchell Stern, Justin Chiu

Website: mt-class.org/penn

CFGs

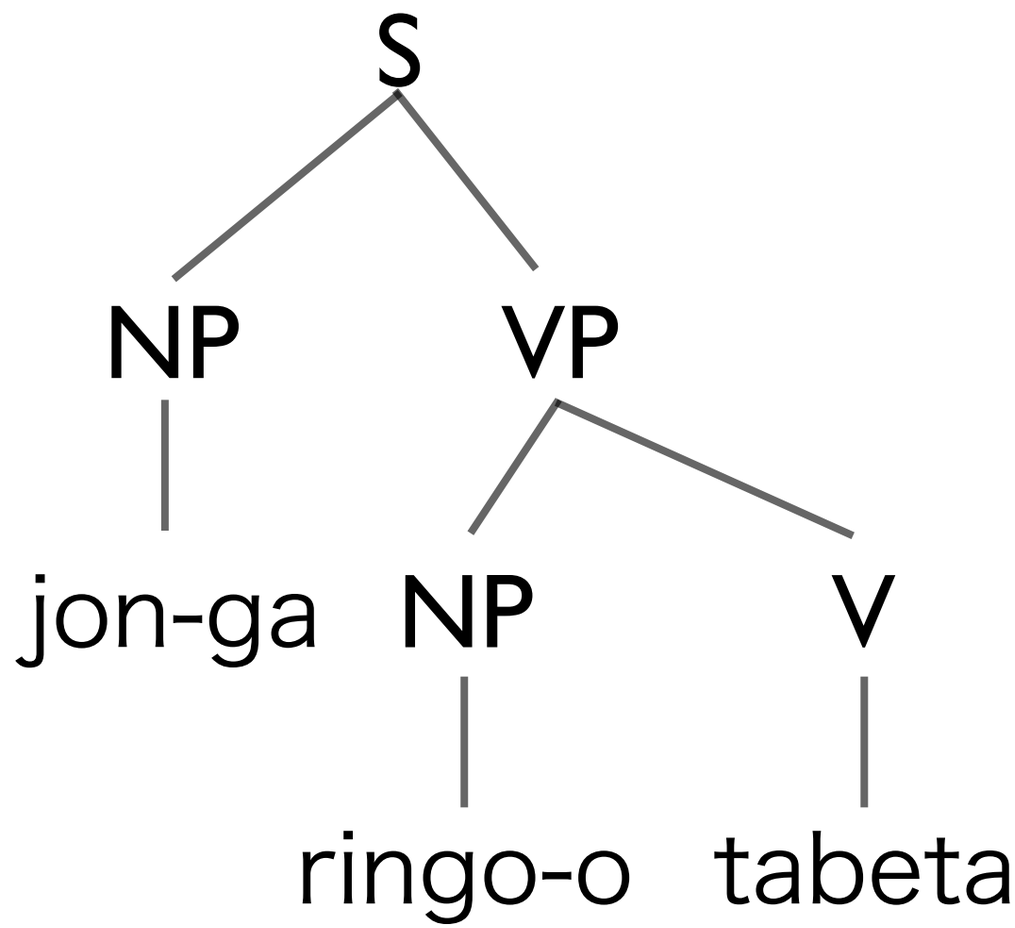
S → NP VP

VP → NP V

V → tabeta

NP → jon-ga

NP → ringo-o



Output: jon-ga ringo-o tabeta

Synchronous CFGs

S → NP VP

VP → NP V

V → tabeta

NP → jon-ga

NP → ringo-o

Synchronous CFGs

S → NP VP : 1 2 (monotonic)

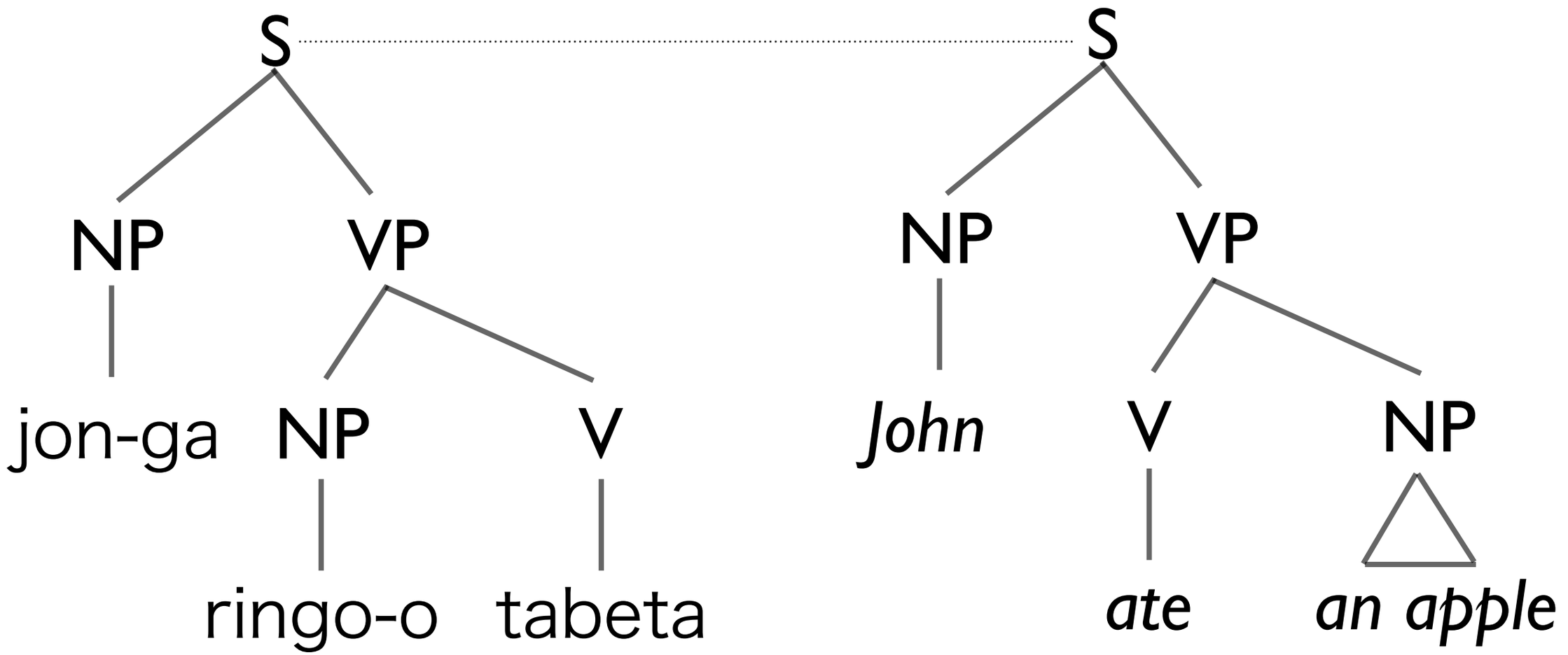
VP → NP V : 2 1 (inverted)

V → tabeta : *ate*

NP → jon-ga : *John*

NP → ringo-o : *an apple*

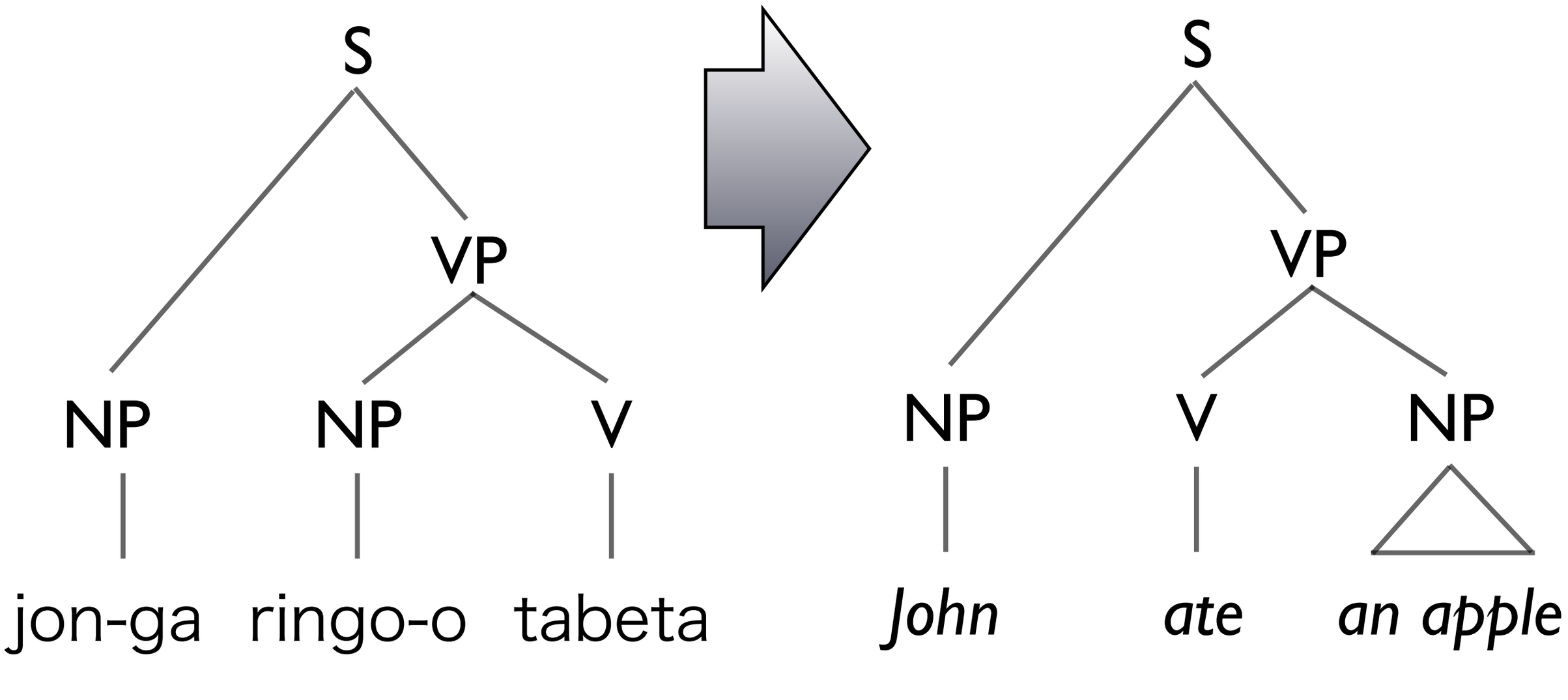
Synchronous generation



Output: (jon-ga ringo-o tabeta : *John ate an apple*)

Translation as parsing

Parse source **Project to target**



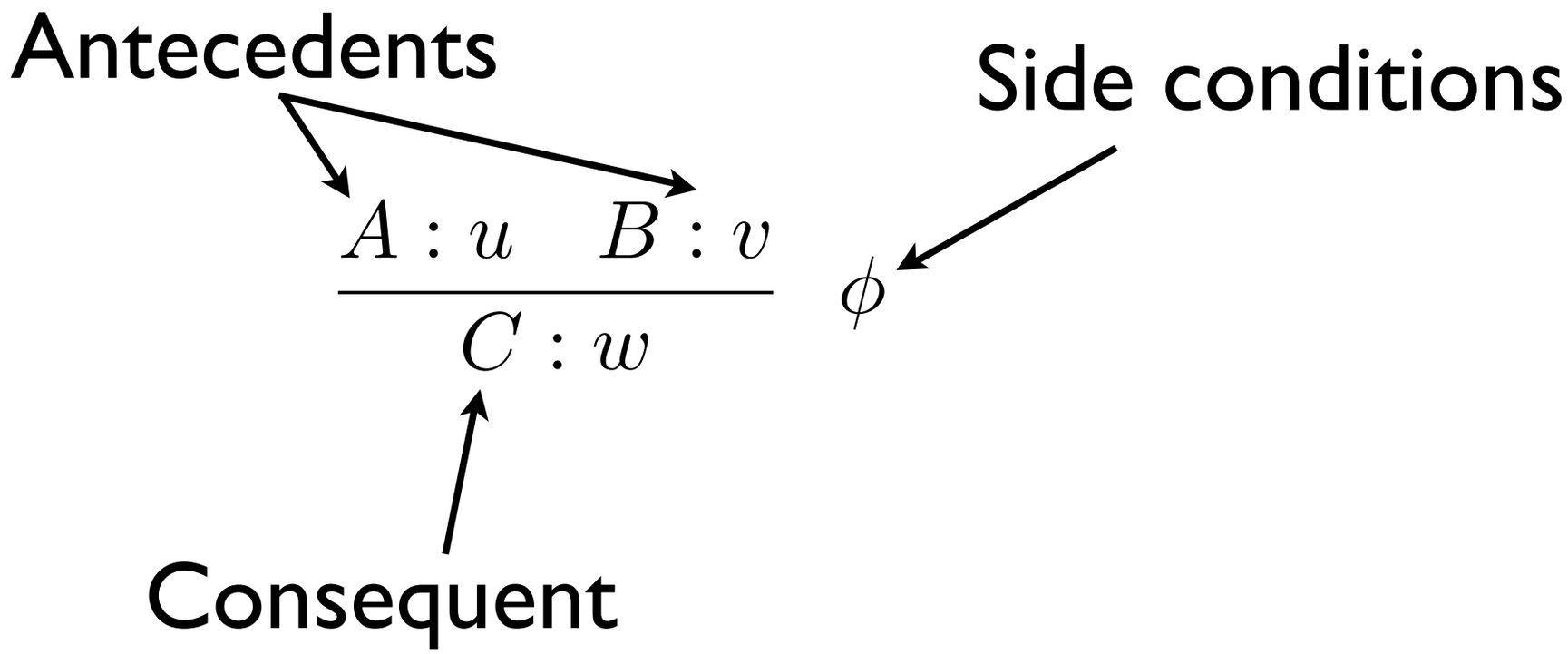
A closer look at parsing

- Parsing is usually done with dynamic programming
  - **Share common computations and structure**
  - Represent exponential number of alternatives in polynomial space
- With SCFGs there are two kinds of ambiguity
  - source parse ambiguity
  - translation ambiguity
  - parse forests can represent both!

A closer look at parsing

- Any monolingual parser can be used (most often: CKY or variants on the CKY algorithm)
- Parsing complexity is O($|n^3|$)
  - cubic in the length of the sentence (n^3)
  - cubic in the number of non-terminals ($|G|^3$)
    - adding nonterminal types increases parsing complexity substantially!
    - With few NTs, exhaustive parsing is tractable

Parsing as deduction



"If A and B are true with weights u and v, and phi is also true, then C is true with weight w."

Example: CKY

Inputs:

$$\mathbf{f} = \langle f_1, f_2, \ldots, f_\ell \rangle$$

G Context-free grammar in Chomsky normal form.

Item form:

$[X, i, j]$ A subtree rooted with NT type X spanning i to j has been recognized.

Example: CKY

Goal:

$$[S, 0, \ell]$$

Axioms:

$$\frac{}{[X, i-1, i] : w} \quad (X \xrightarrow{w} f_i) \in G$$

Inference rules:

$$\frac{[X, i, k] : u \quad [Y, k, j] : v}{[Z, i, j] : u \times v \times w} \quad (Z \xrightarrow{w} XY) \in G$$

S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

V → saw

NN → duck

V → duck

PRP → I

PRP → her

I **saw** **her** **duck**

0 1 2 3 4

S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

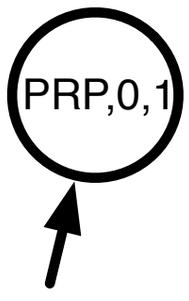
V → saw

NN → duck

V → duck

PRP → I

PRP → her



I **saw** **her** **duck**

0 1 2 3 4

S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

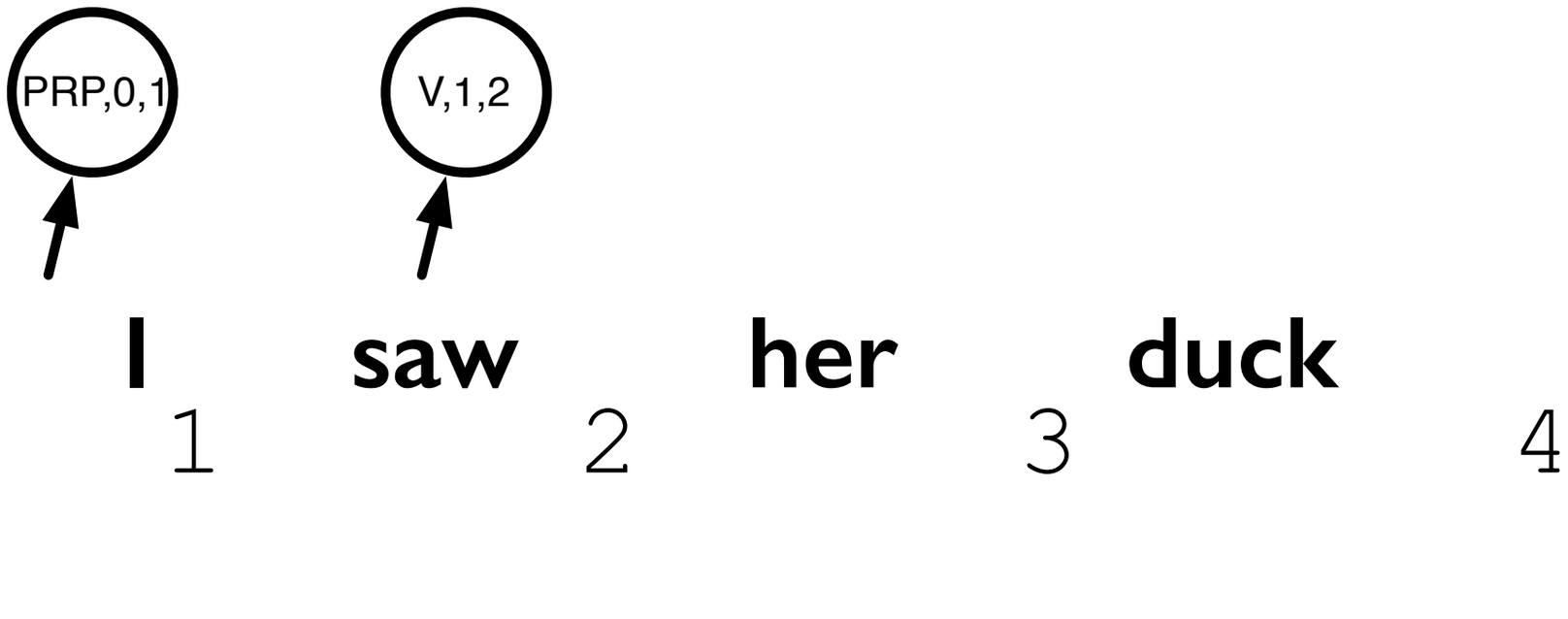
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

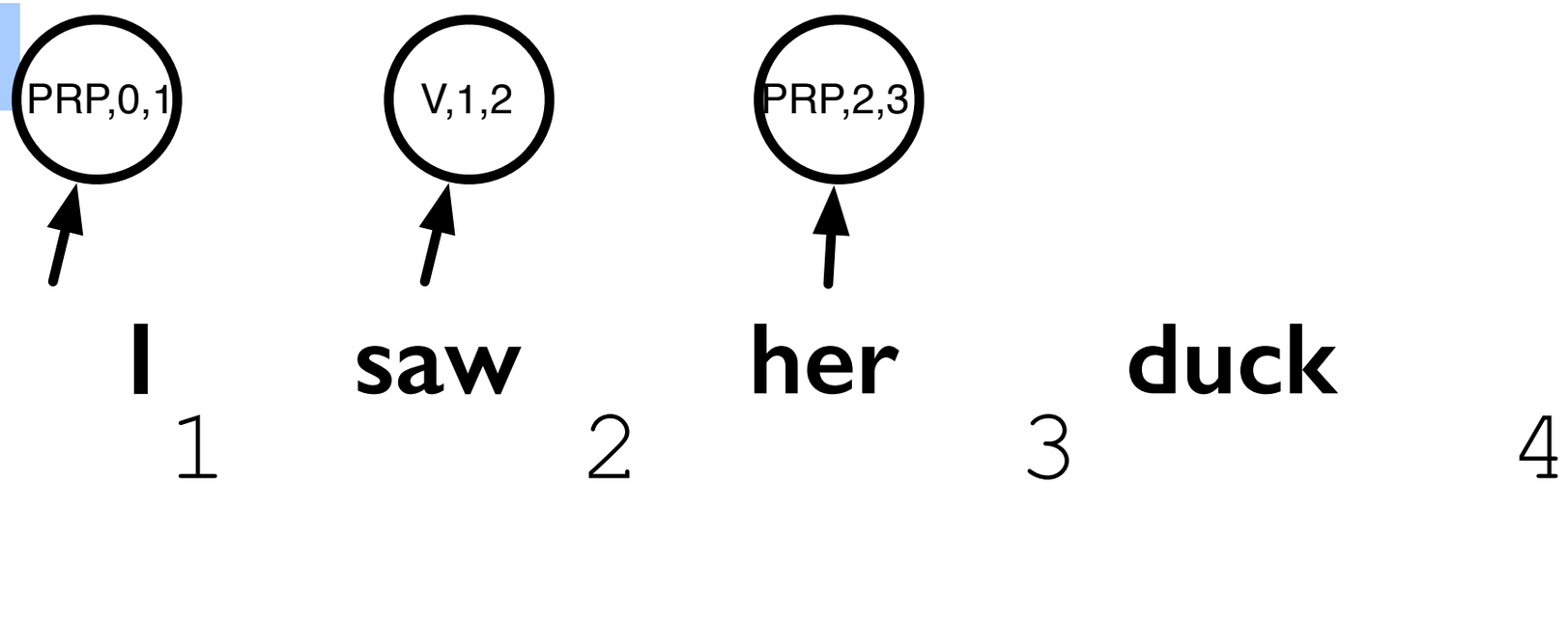
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

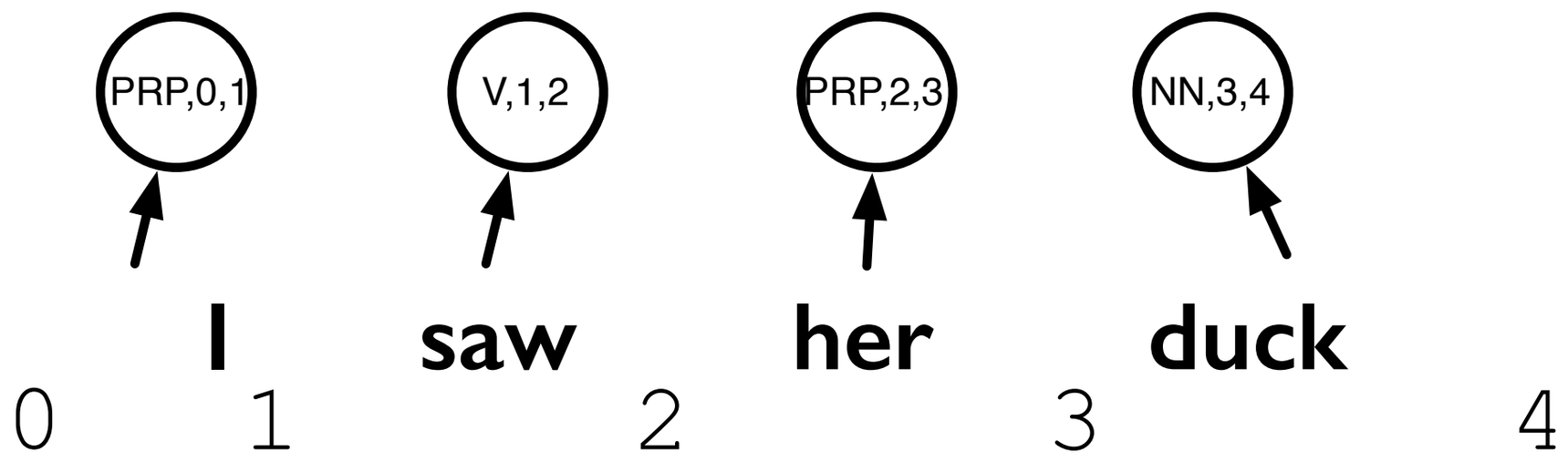
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

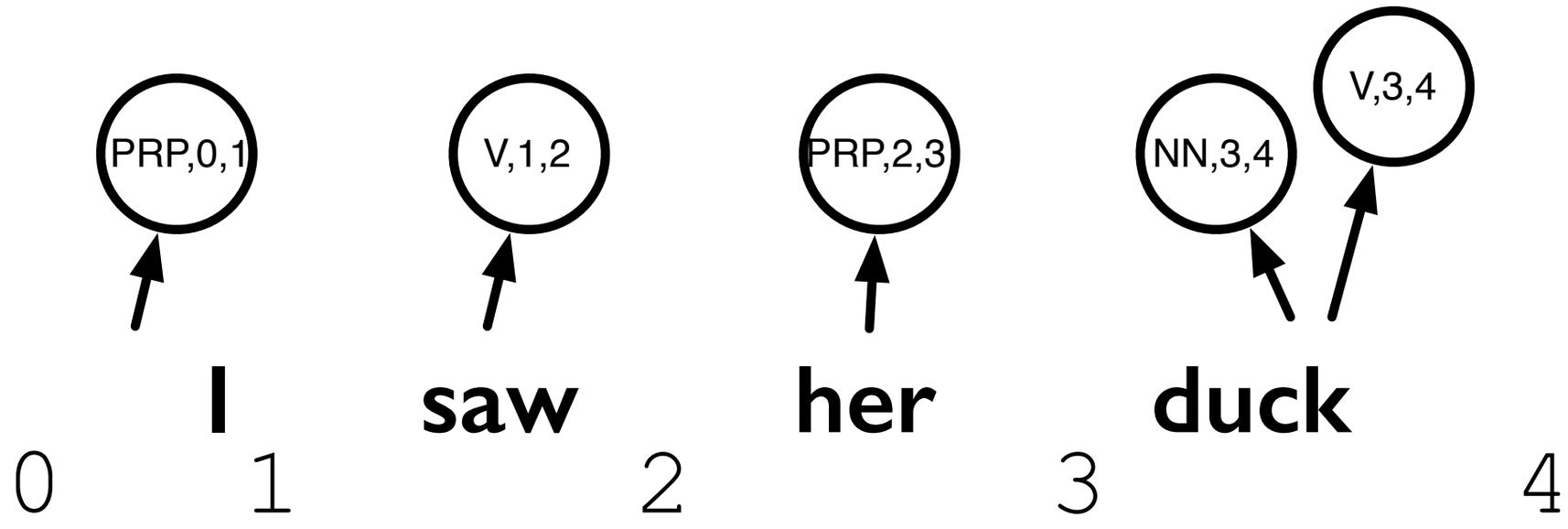
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

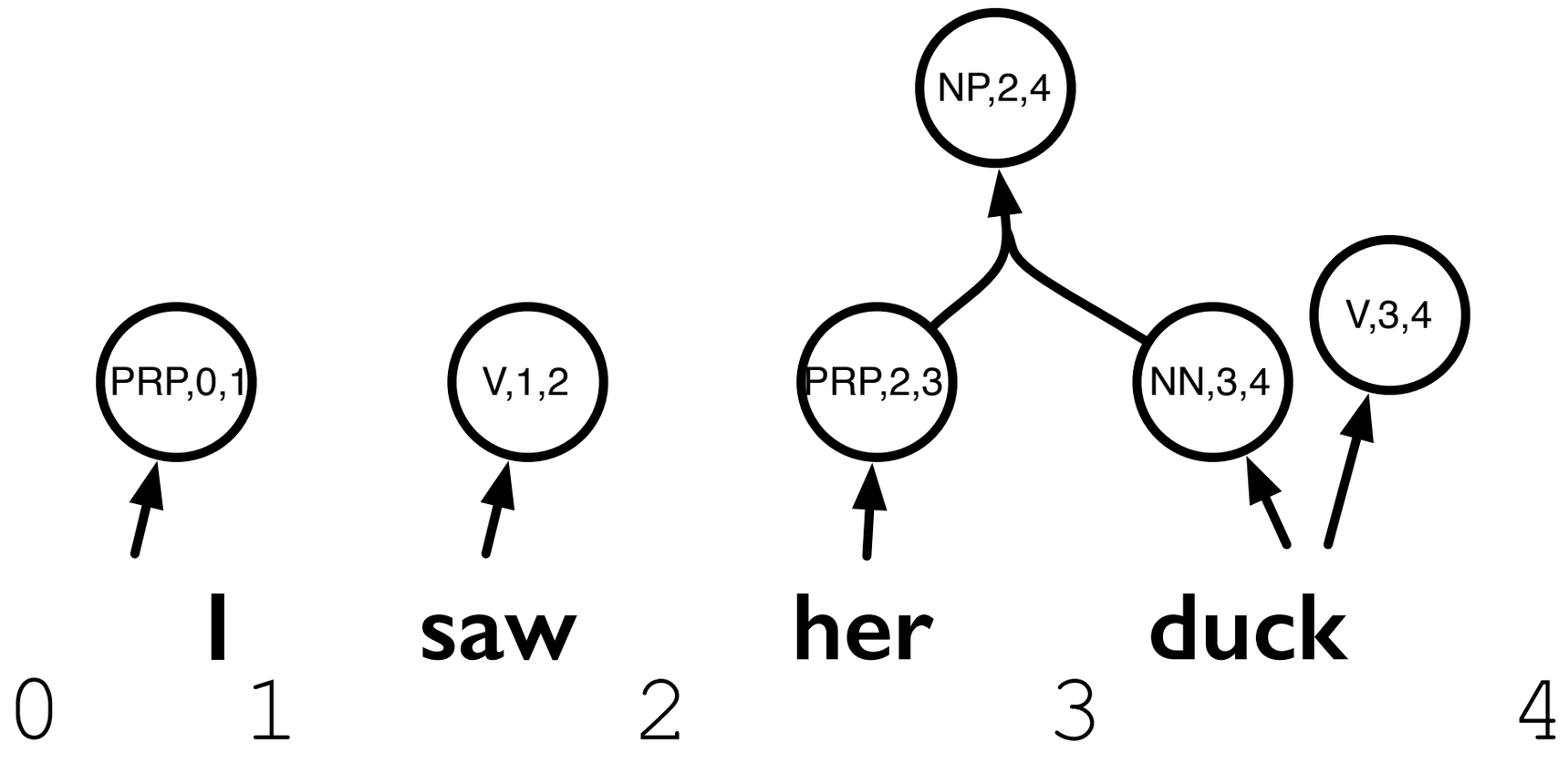
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

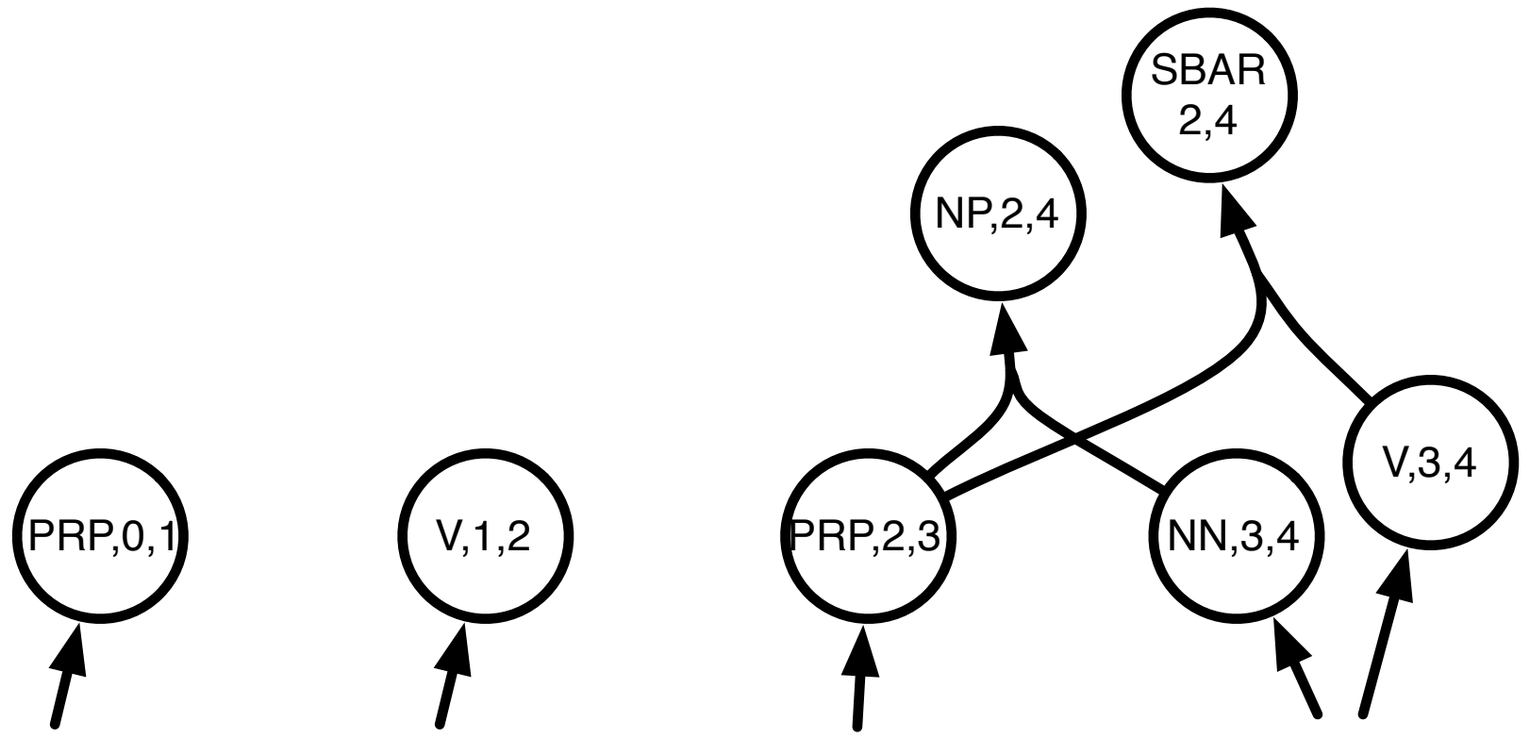
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

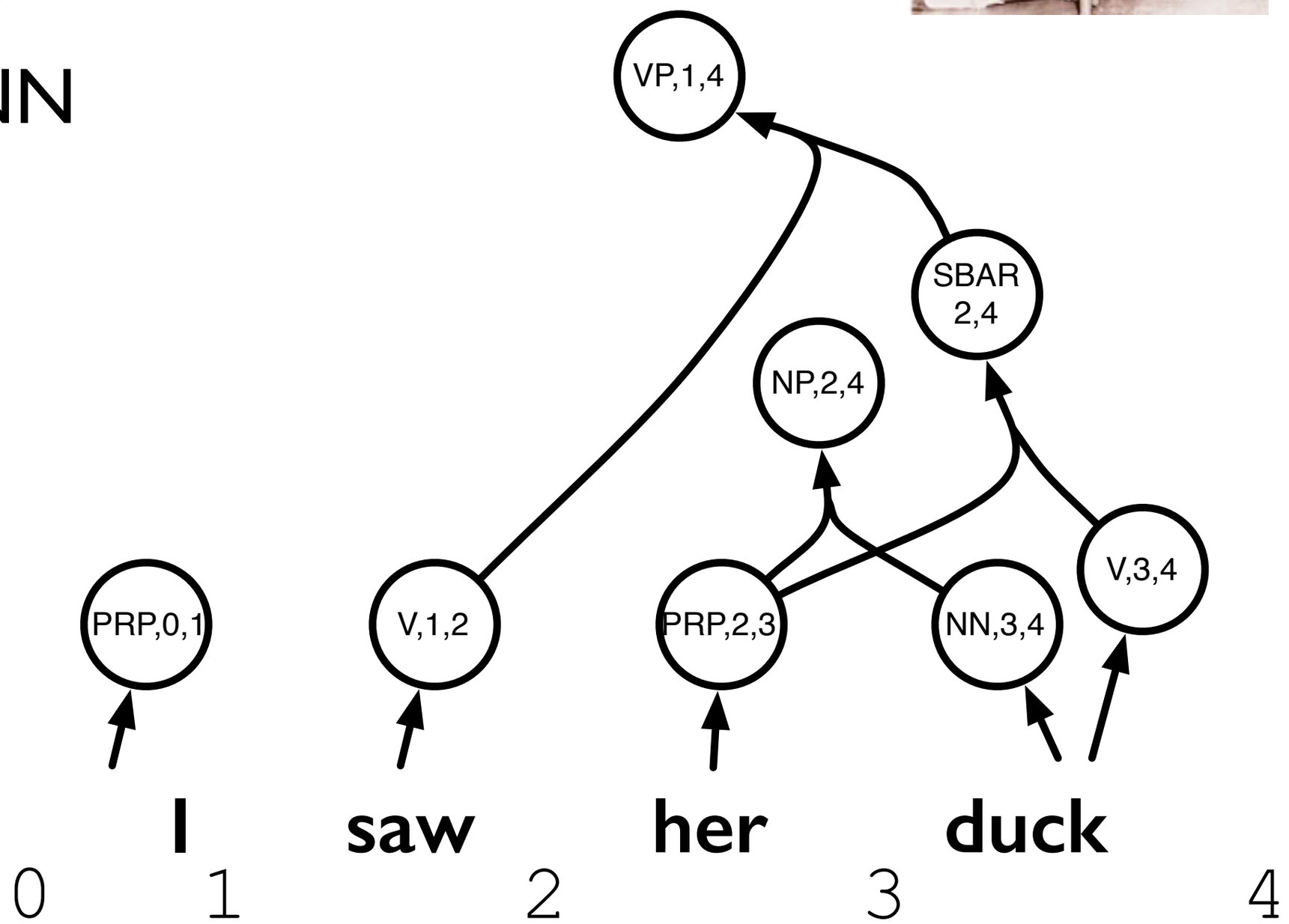
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

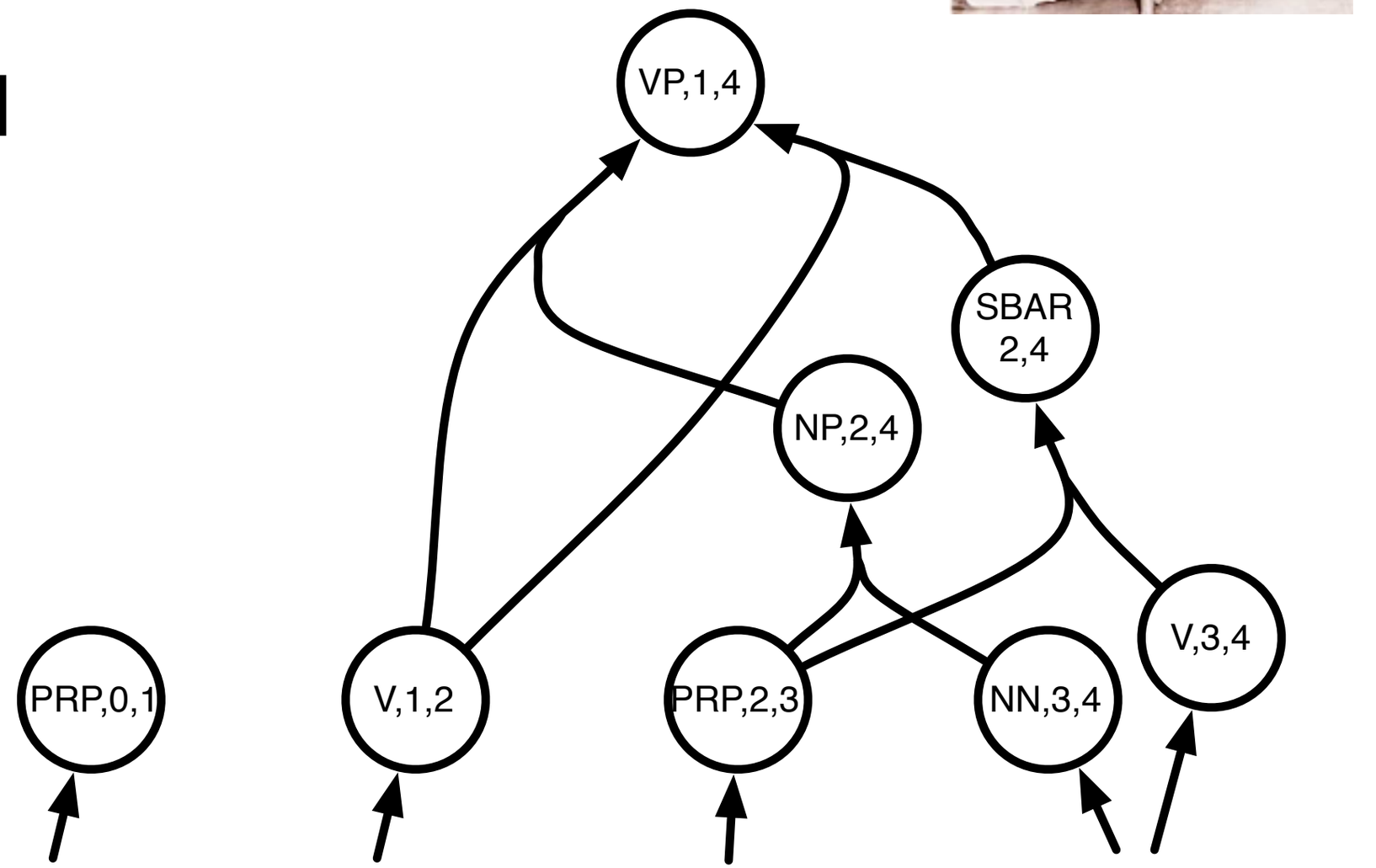
V → saw

NN → duck

V → duck

PRP → I

PRP → her



S → PRP VP

VP → V NP

VP → V SBAR

SBAR → PRP V

NP → PRP NN

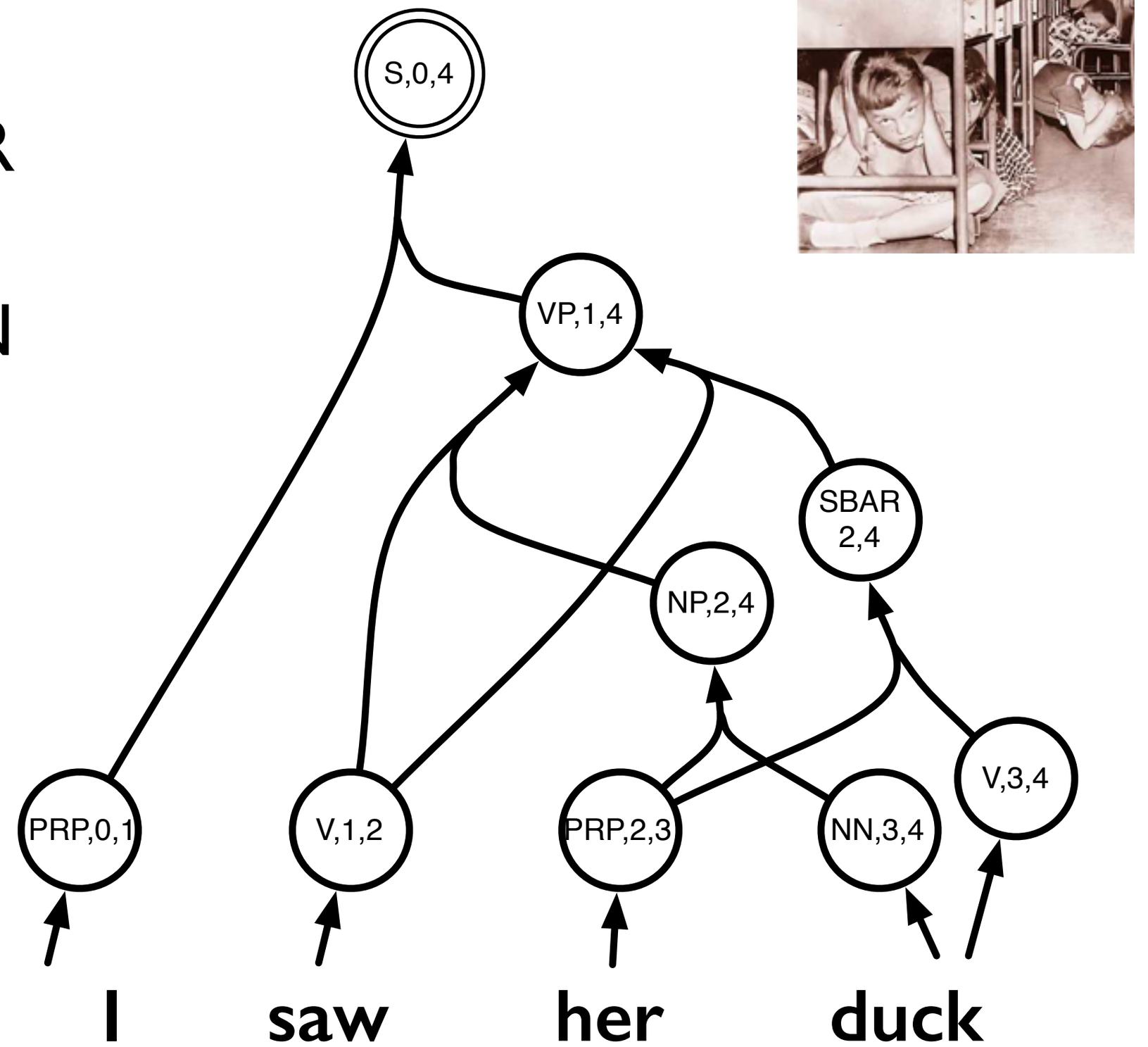
V → saw

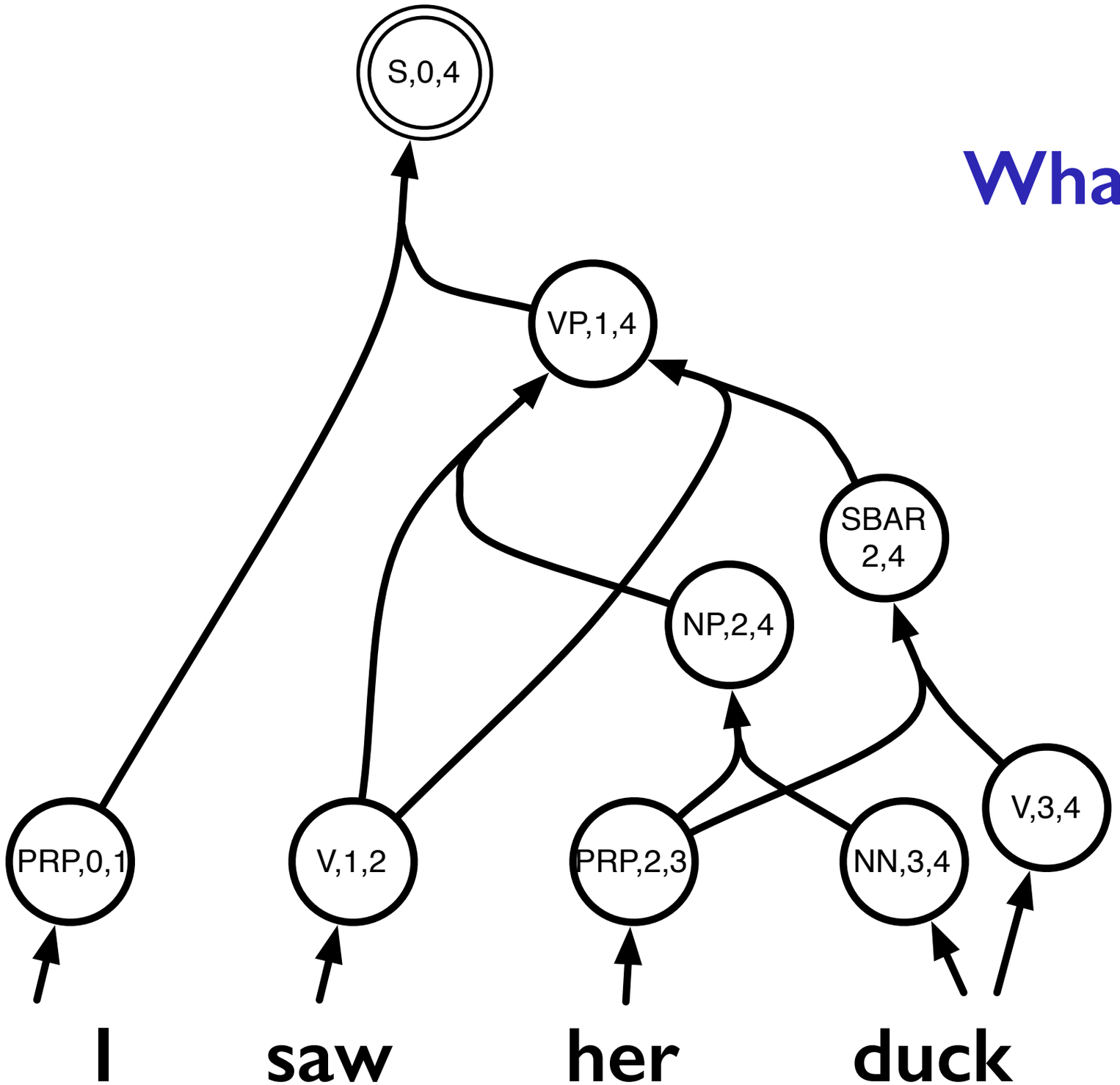
NN → duck

V → duck

PRP → I

PRP → her





What is this object?

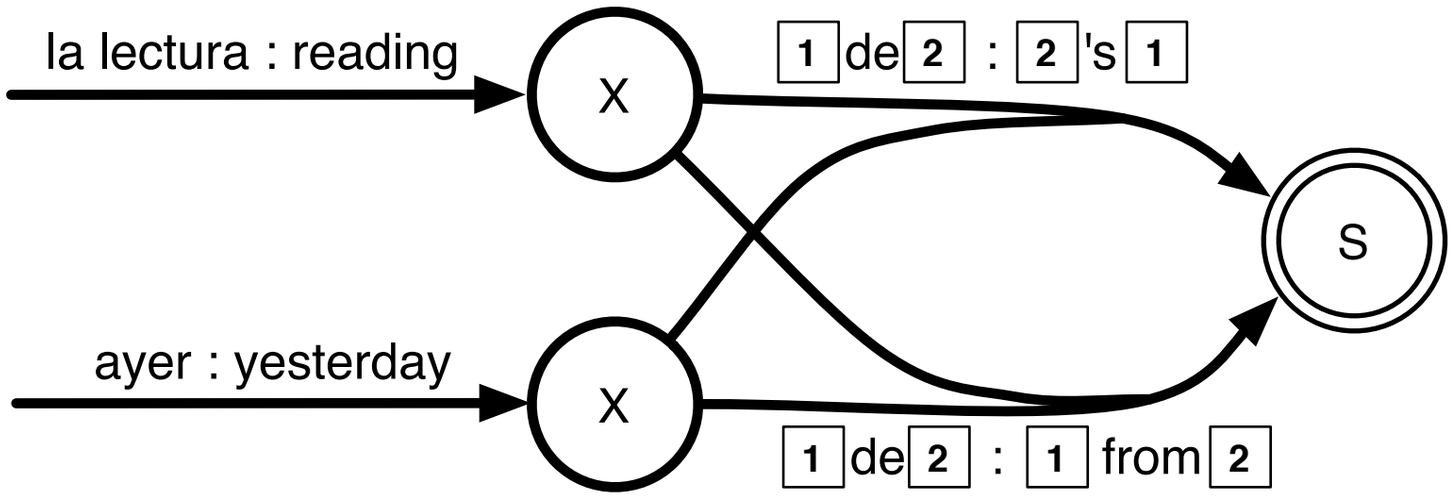
Semantics of hypergraphs

- Generalization of directed graphs
- Special node designated the "goal"
- Every edge has a single head and 0 or more tails (the **arity** of the edge is the number of tails)
- Node labels correspond to LHS's of CFG rules
- A **derivation** is the generalization of the graph concept of **path** to hypergraphs
- Weights multiply along edges in the derivation, and add at nodes (cf. **semiring parsing**)

Edge labels

- Edge labels may be a mix of terminals and substitution sites (non-terminals)
- In translation hypergraphs, edges are labeled in both the source and target languages
- The number of substitution sites must be equal to the arity of the edge and must be the same in both languages
- The two languages may have different orders of the substitution sites
- There is no restriction on the number of terminal symbols

Edge labels



{(la lectura de ayer : *yesterday 's reading*),
(la lectura de ayer : *reading from yesterday*)}

Inference algorithms

- Viterbi $O(|E| + |V|)$
  - Find the maximum weighted derivation
  - Requires a partial ordering of weights
- Inside - outside $O(|E| + |V|)$
  - Compute the marginal (sum) weight of all derivations passing through each edge/node
- k-best derivations $O(|E| + |D_{max}|k \log k)$
  - Enumerate the k-best derivations in the hypergraph
  - See IWPT paper by Huang and Chiang (2005)

Things to keep in mind

Bound on the number of edges:

$$|E| \in O(n^3|G|^3)$$

Bound on the number of nodes:

$$|V| \in O(n^2|G|)$$

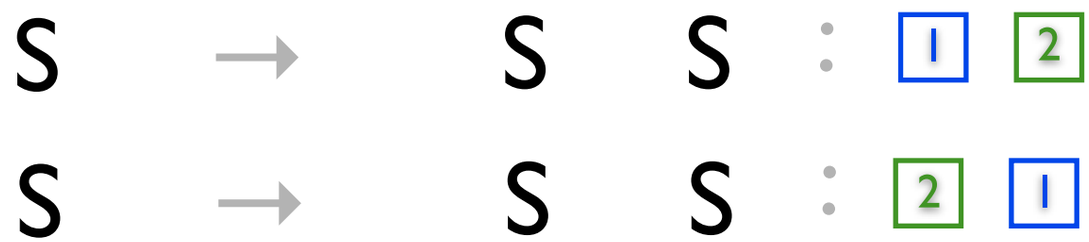
Decoding Again

- Translation hypergraphs are a "lingua franca" for translation search spaces
  - Note that FST lattices are a special case
- **Decoding problem: how do I build a translation hypergraph?**

Representational limits

Consider this very simple SCFG translation model:

"Glue" rules:



Representational limits

Consider this very simple SCFG translation model:

"Glue" rules:

S → S S : 1 2

S → S S : 2 1

"Lexical" rules:

S → tabeta : *ate*

S → jon-ga : *John*

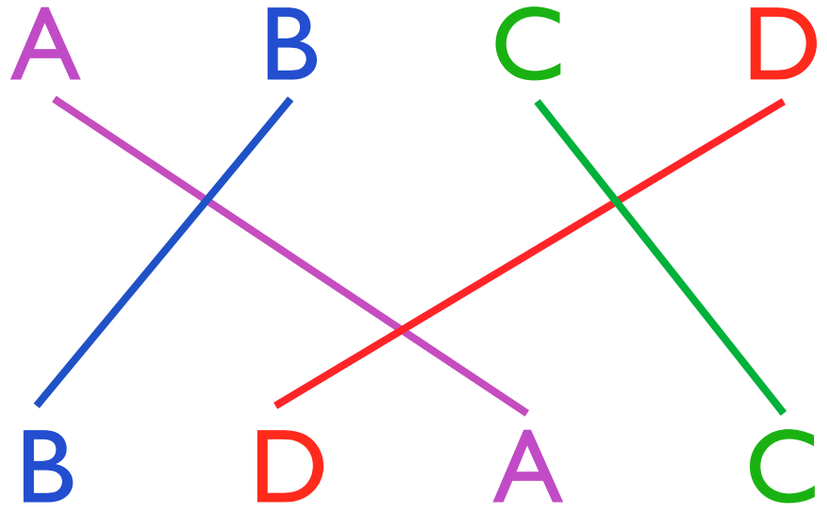
S → ringo-o : *an apple*

Representational limits

- Phrase-based decoding runs in exponential time
  - All permutations of the source are modeled (traveling salesman problem!)
  - Typically distortion limits are used to mitigate this
- But parsing is polynomial...what's going on?

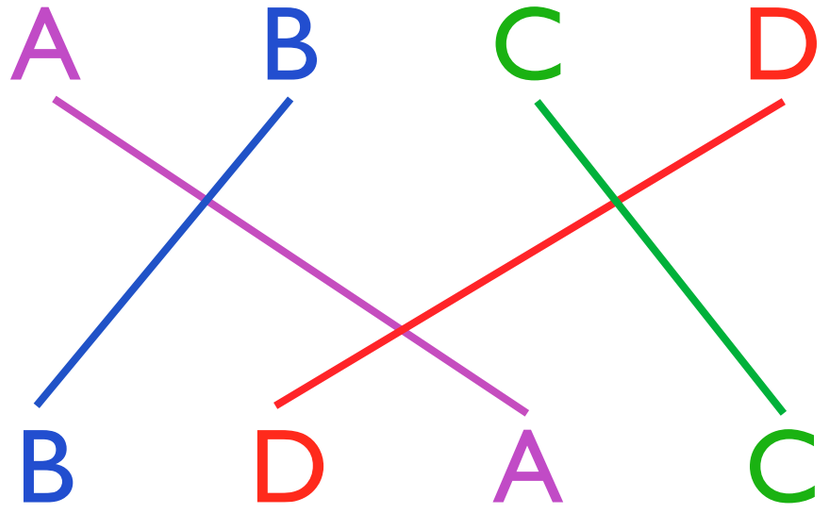
Representational limits

Binary SCFGs cannot model this
(however, ternary SCFGs can):



Representational limits

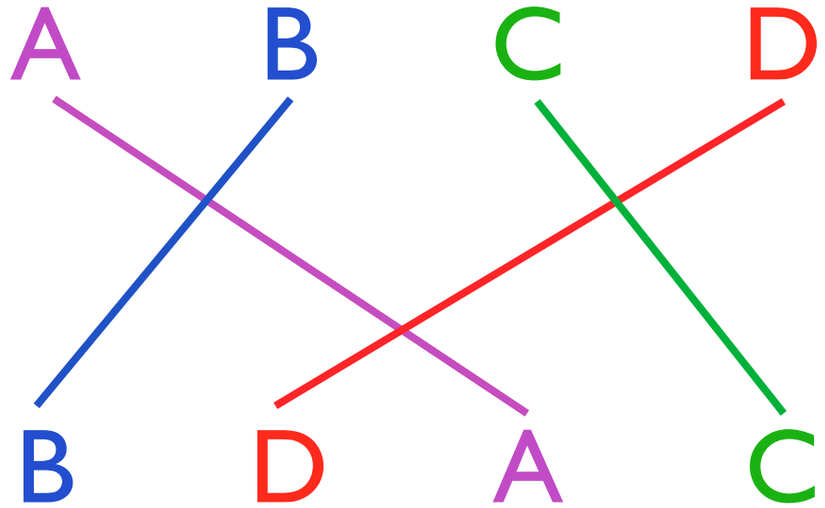
Binary SCFGs cannot model this
(however, ternary SCFGs can):



But can't we binarize *any* grammar?

Representational limits

Binary SCFGs cannot model this
(however, ternary SCFGs can):



But can't we binarize *any* grammar?

No. Synchronous CFGs cannot generally be binarized!

Does this matter?

- The "forbidden" pattern is observed in real data (Melamed, 2003)
- Does this matter?
  - Learning
    - Phrasal units and higher rank grammars can account for the pattern
    - Sentences can be simplified or ignored
  - Translation
    - The pattern does exist, but how often **must** it exist (i.e., is there a good translation that doesn't violate the SCFG matching property)?

Tree-to-string

- How do we generate a hypergraph for a tree-to-string translation model?
  - Simple linear-time (given a fixed translation model) top-down matching algorithm
    - Recursively cover "uncovered" sites in tree
- Each node in the input tree becomes a node in the translation forest
- For details, Huang et al. (AMTA, 2006) and Huang et al. (EMNLP, 2010)

$$\mathrm{S}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{VP}) \rightarrow x_1\ x_2$$

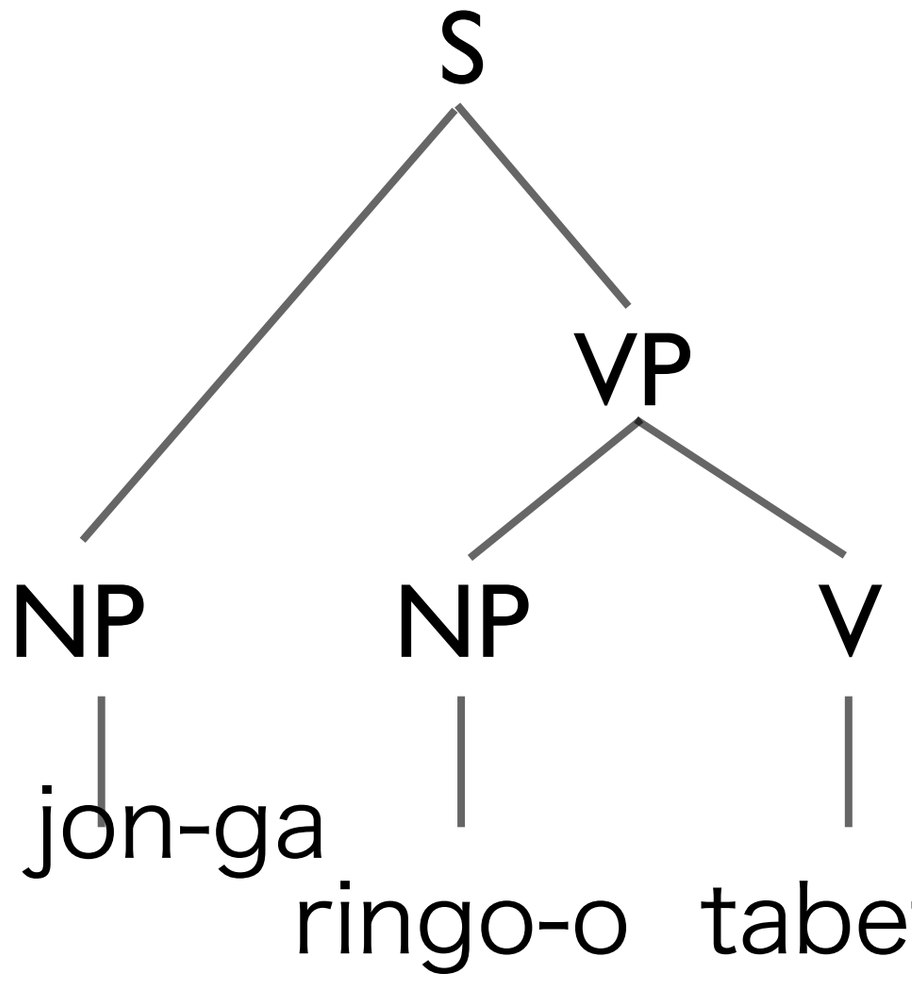
$$\mathrm{VP}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{V}) \rightarrow x_2\ x_1$$

$$tabeta \rightarrow ate$$

$$ringo\text{-}o \rightarrow an\ apple$$

$$jon\text{-}ga \rightarrow John$$

} Tree-to-string grammar



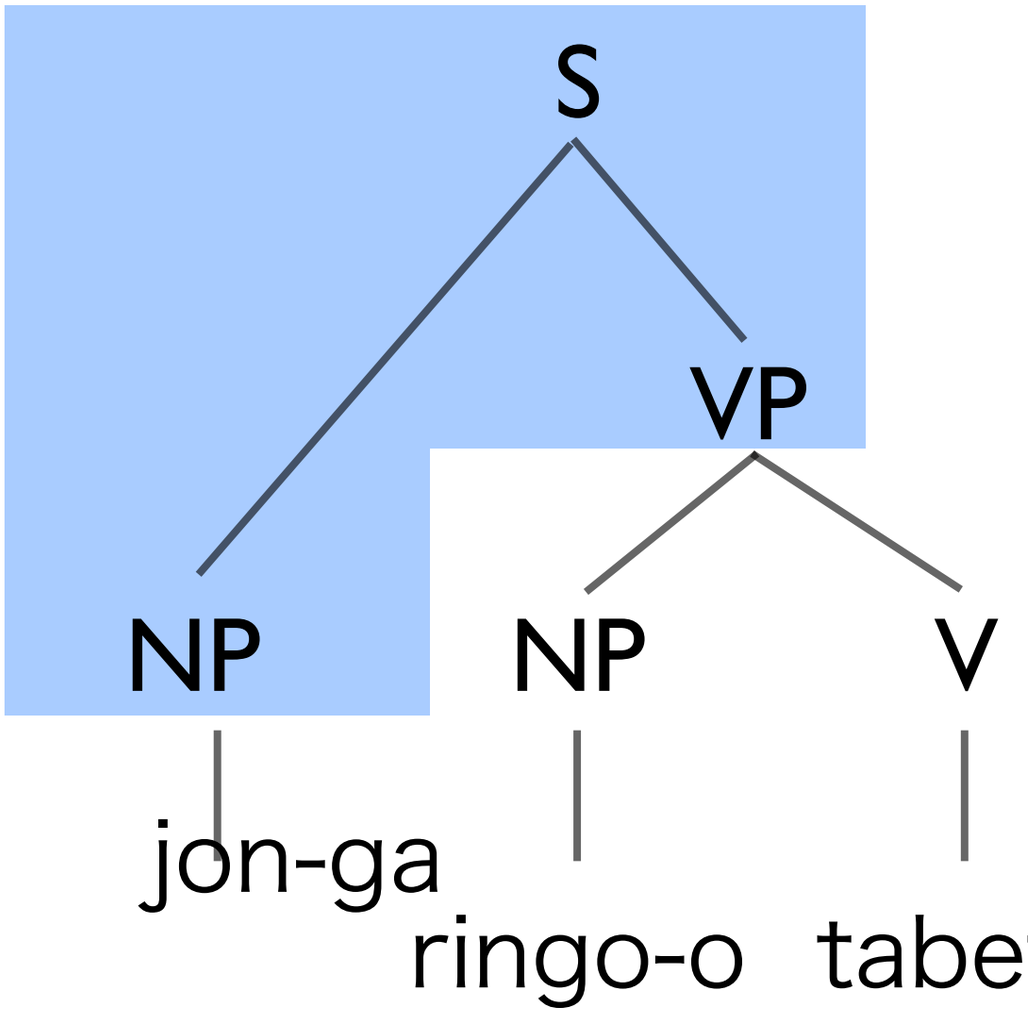
$$\mathrm{S}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{VP}) \rightarrow x_1\ x_2$$

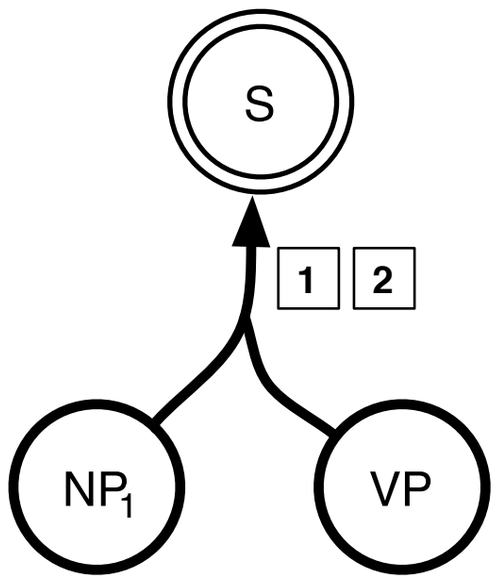
$$\mathrm{VP}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{V}) \rightarrow x_2\ x_1$$

$$tabeta \rightarrow ate$$

$$ringo\text{-}o \rightarrow an\ apple$$

$$jon\text{-}ga \rightarrow John$$





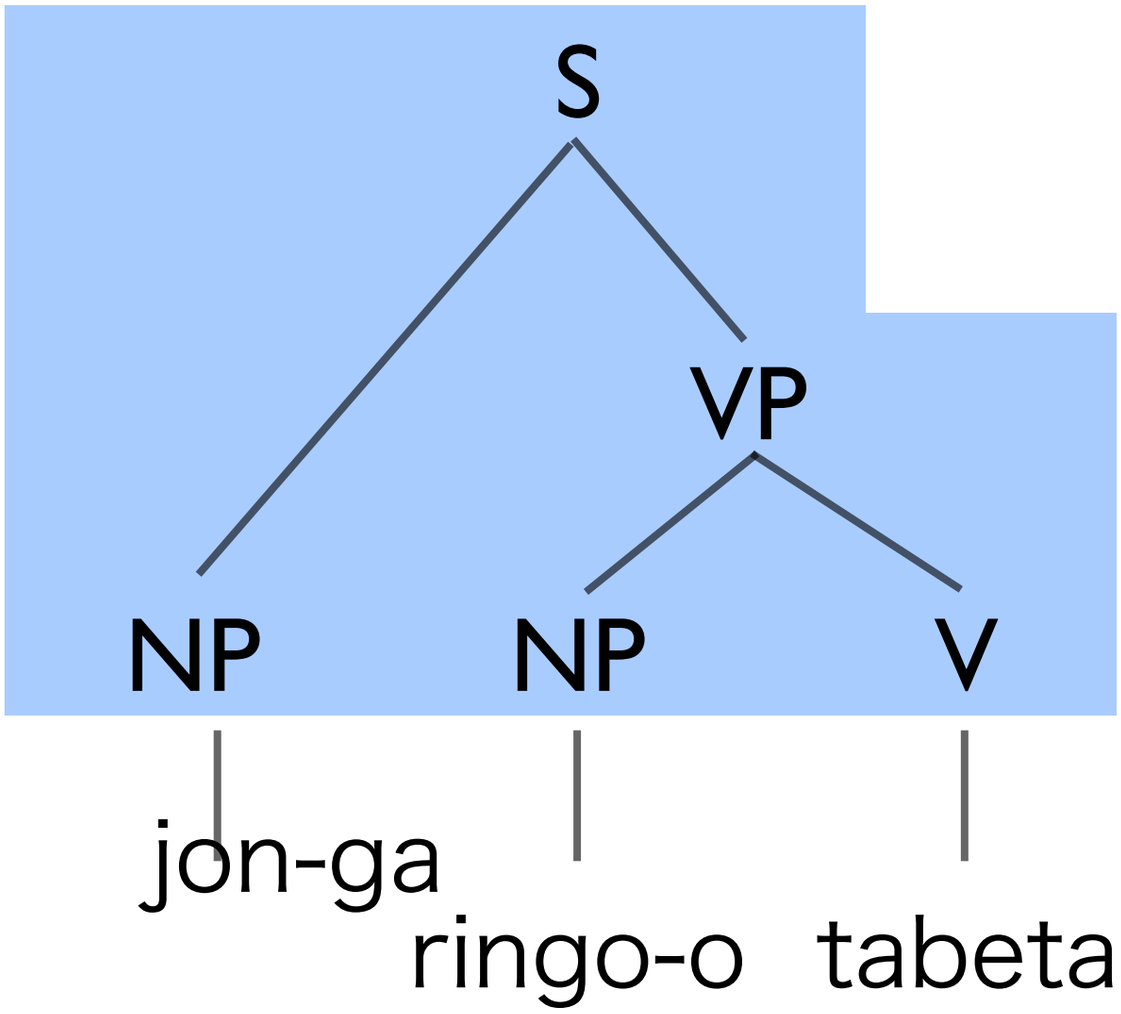
$$\mathrm{S}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{VP}) \rightarrow x_1\ x_2$$

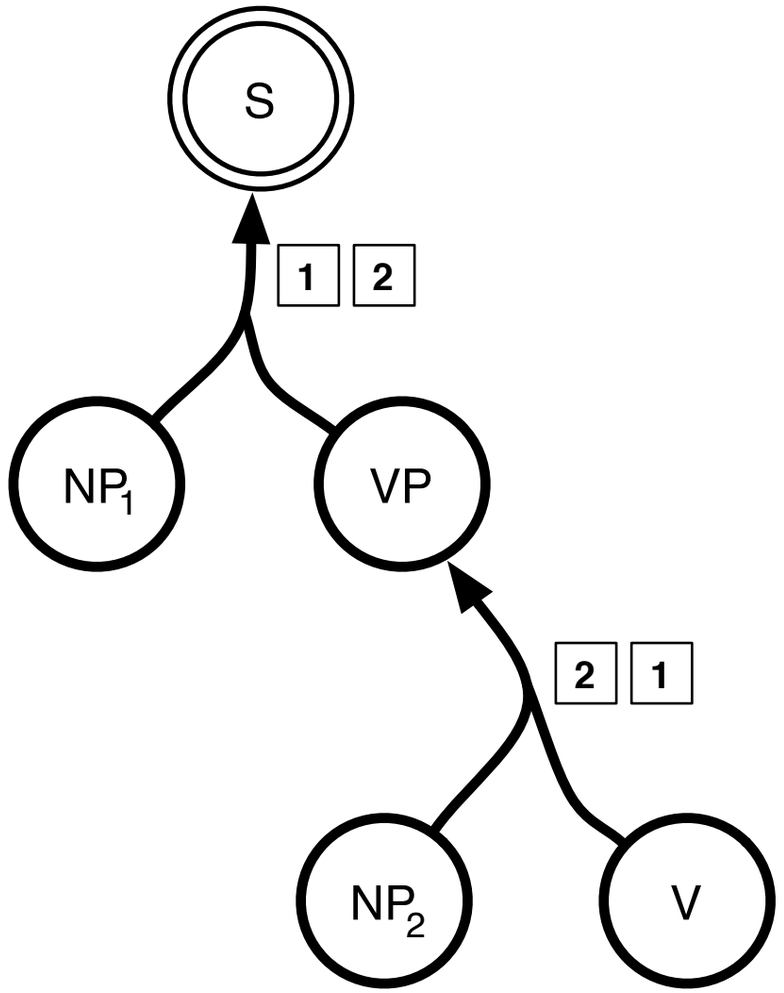
$$\mathrm{VP}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{V}) \rightarrow x_2\ x_1$$

$$tabeta \rightarrow ate$$

$$ringo\text{-}o \rightarrow an\ apple$$

$$jon\text{-}ga \rightarrow John$$





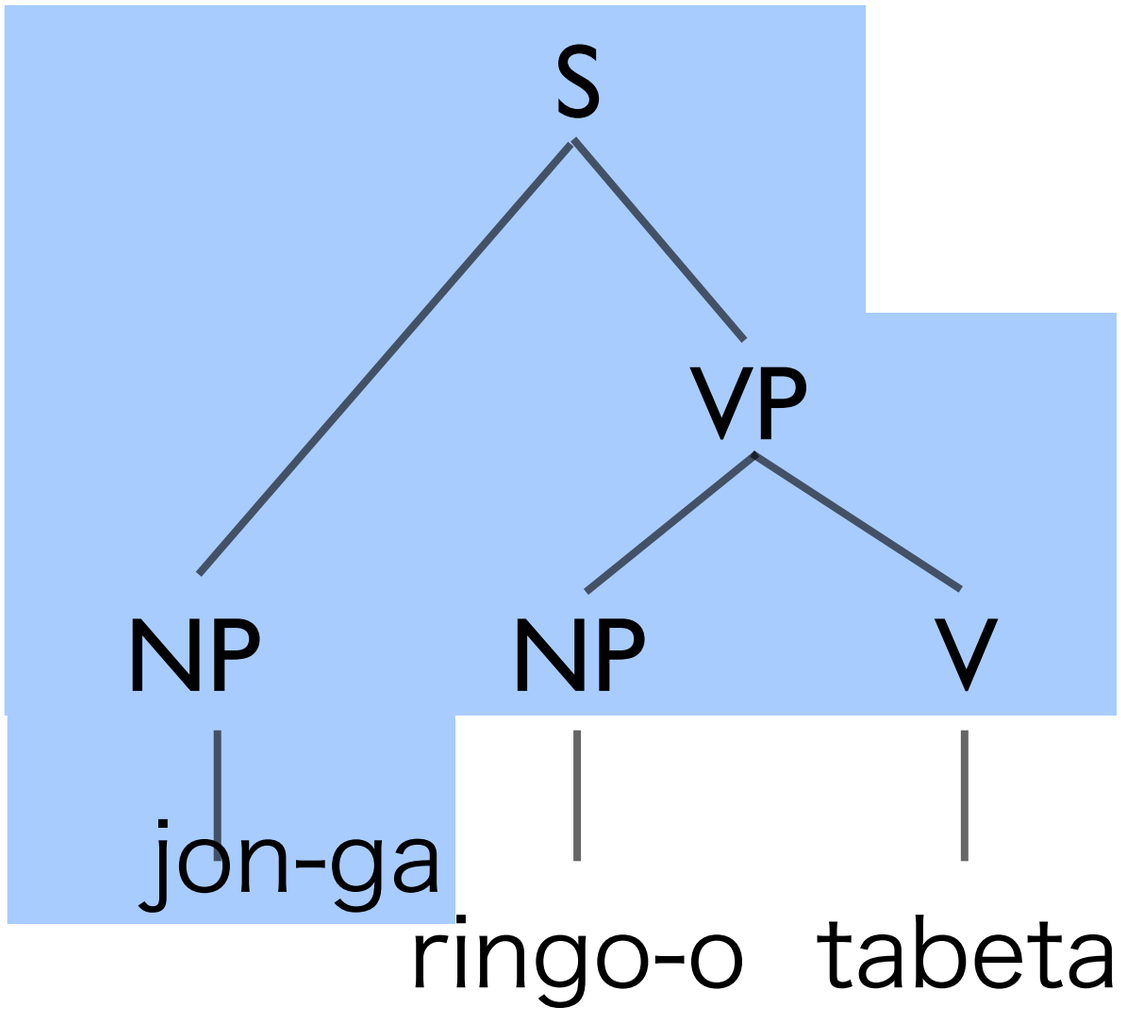
$$\mathrm{S}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{VP}) \rightarrow x_1\ x_2$$

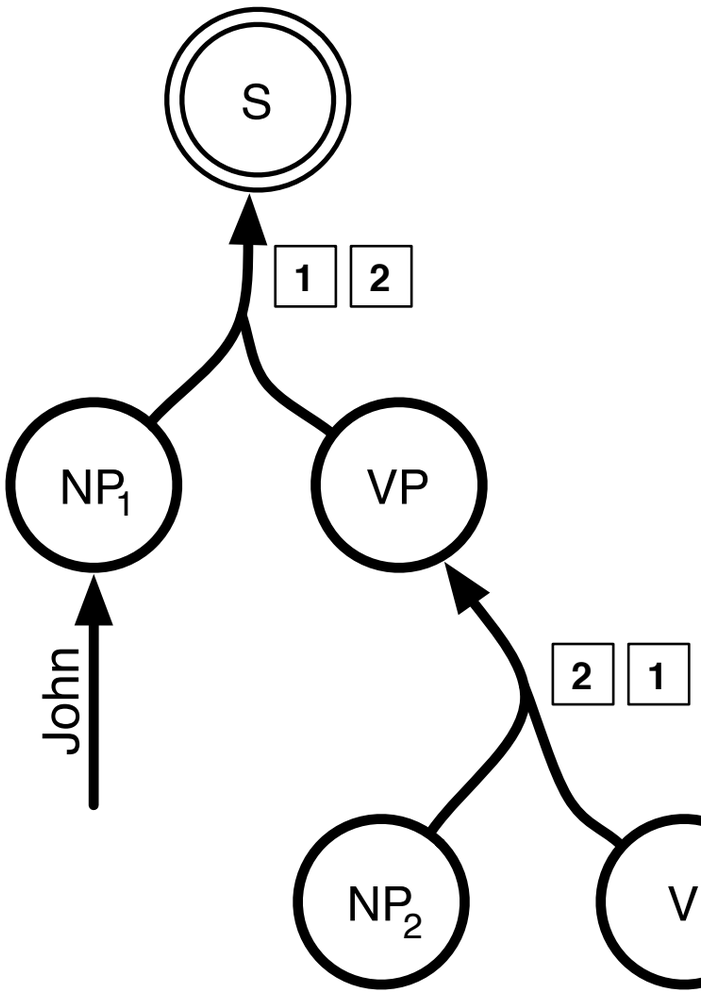
$$\mathrm{VP}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{V}) \rightarrow x_2\ x_1$$

$$tabeta \rightarrow ate$$

$$ringo\text{-}o \rightarrow an\ apple$$

$$jon\text{-}ga \rightarrow John$$





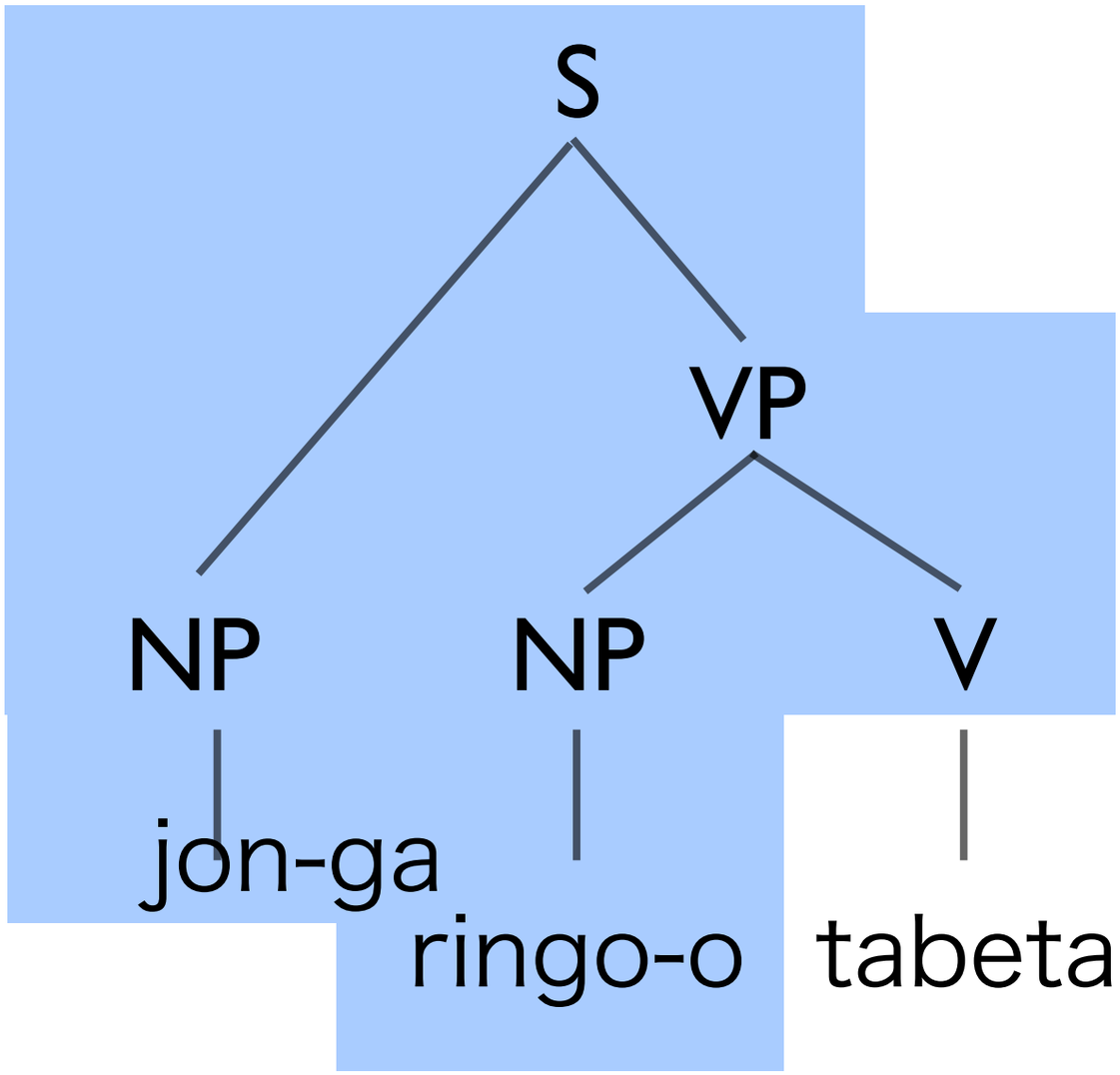
$$\mathrm{S}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{VP}) \rightarrow x_1\ x_2$$

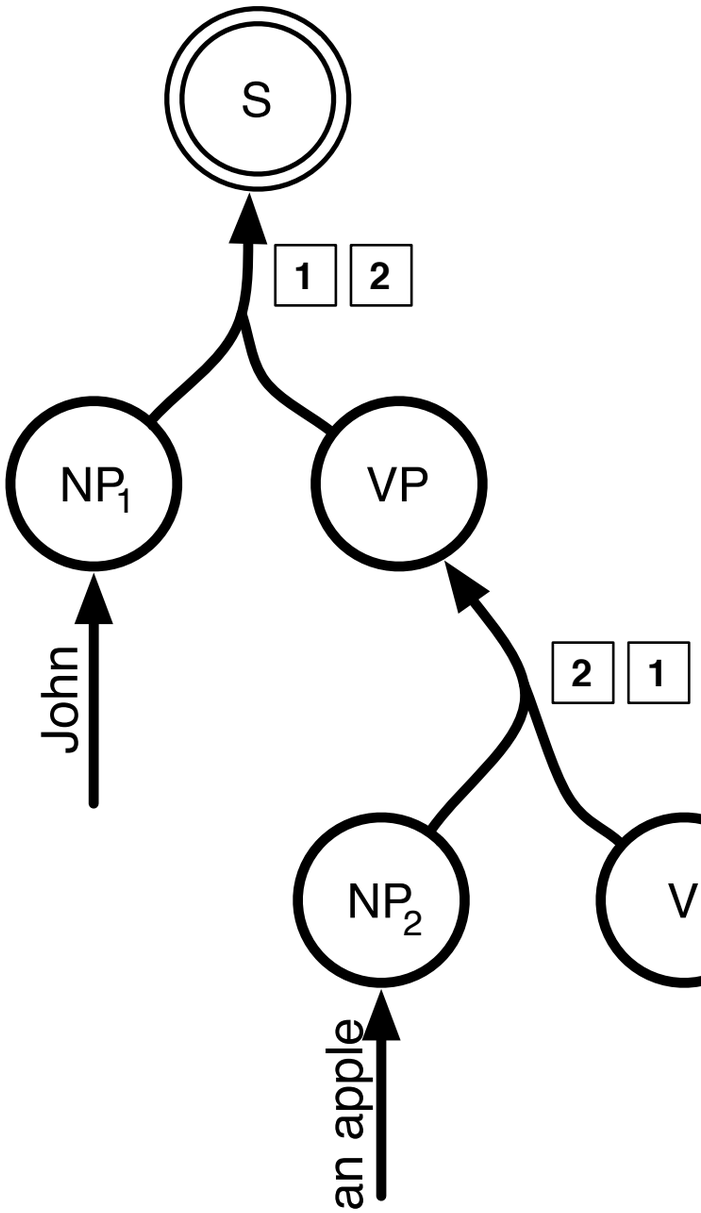
$$\mathrm{VP}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{V}) \rightarrow x_2\ x_1$$

$$tabeta \rightarrow ate$$

$$ringo\text{-}o \rightarrow an\ apple$$

$$jon\text{-}ga \rightarrow John$$





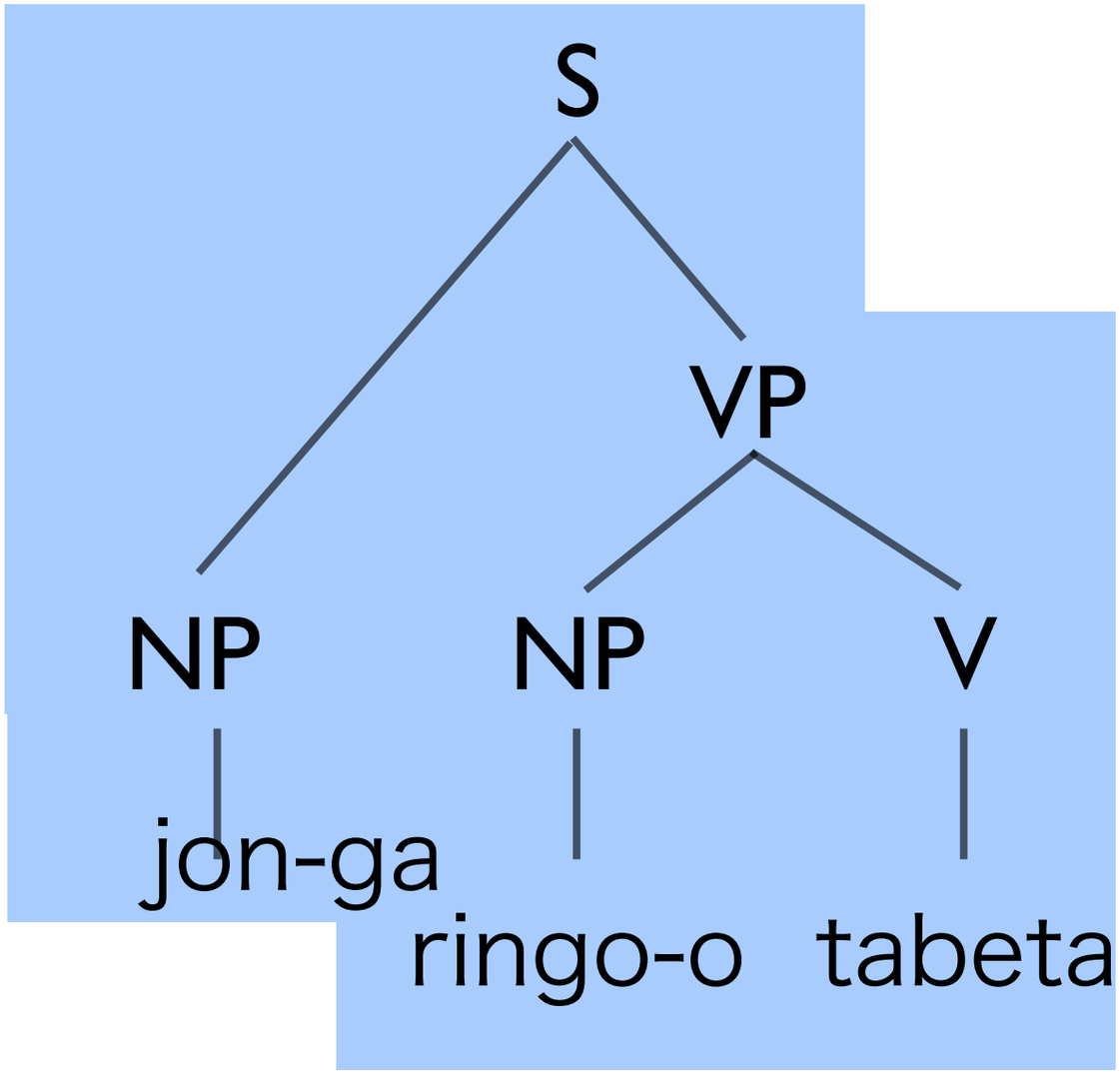
$$\mathrm{S}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{VP}) \rightarrow x_1\ x_2$$

$$\mathrm{VP}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{V}) \rightarrow x_2\ x_1$$

$$tabeta \rightarrow ate$$

$$ringo\text{-}o \rightarrow an\ apple$$

$$jon\text{-}ga \rightarrow John$$



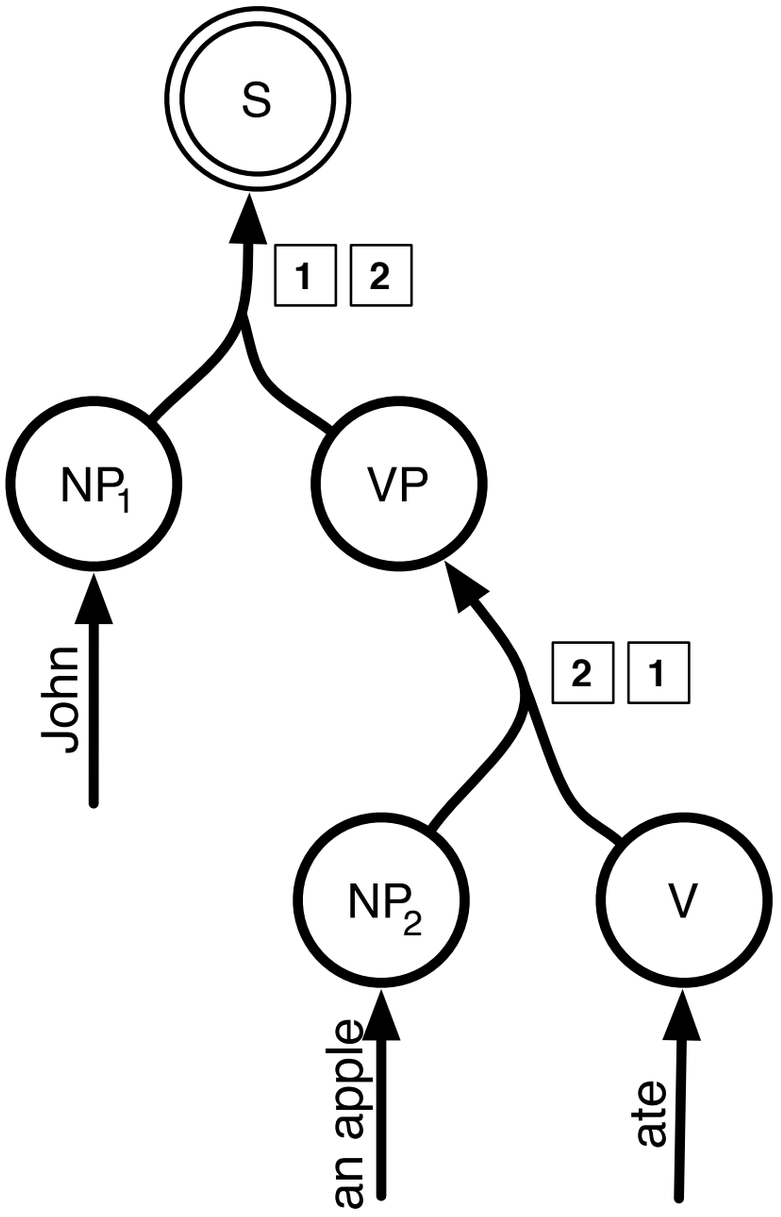
$$\mathrm{S}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{VP}) \rightarrow x_1\ x_2$$

$$\mathrm{VP}(x_1{:}\mathrm{NP}\ x_2{:}\mathrm{V}) \rightarrow x_2\ x_1$$

$$tabeta \rightarrow ate$$

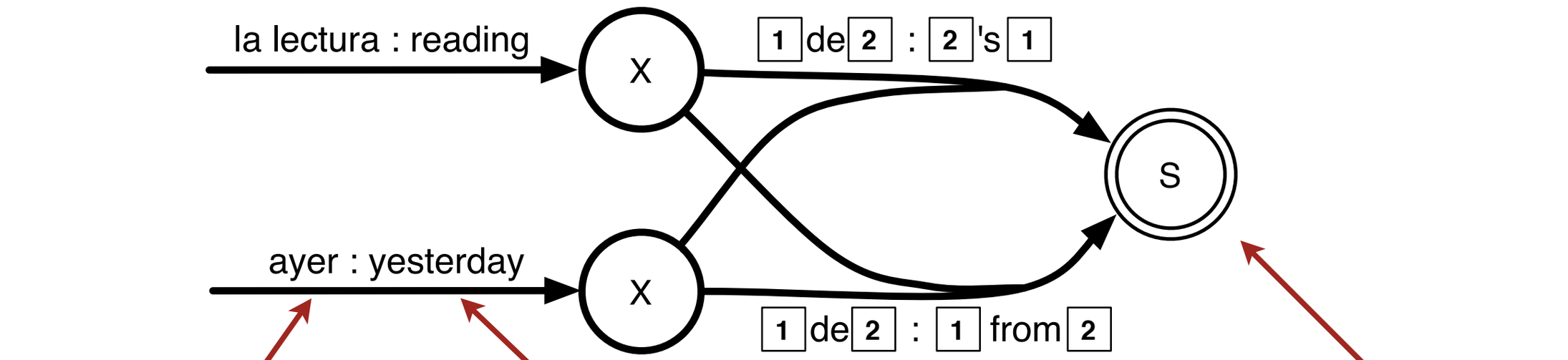
$$ringo\text{-}o \rightarrow an\ apple$$

$$jon\text{-}ga \rightarrow John$$



Language Models

Hypergraph review

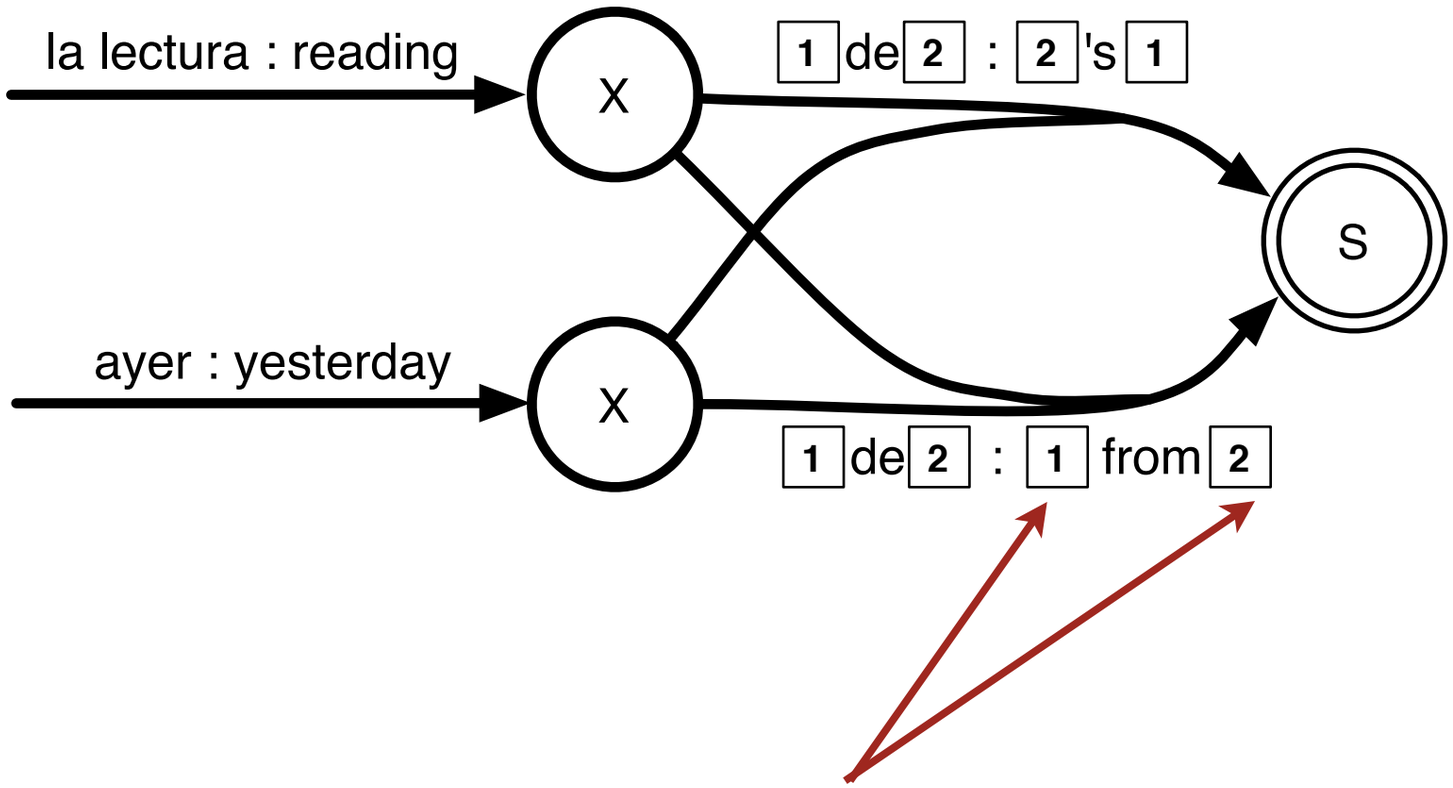


Source label

Target label

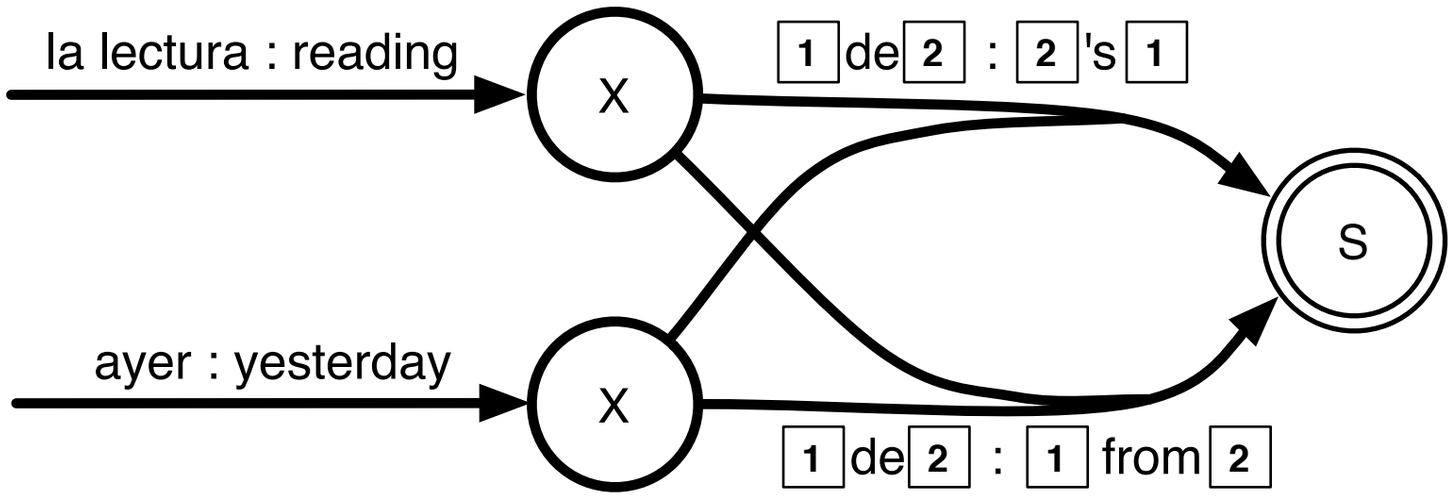
Goal node

Hypergraph review



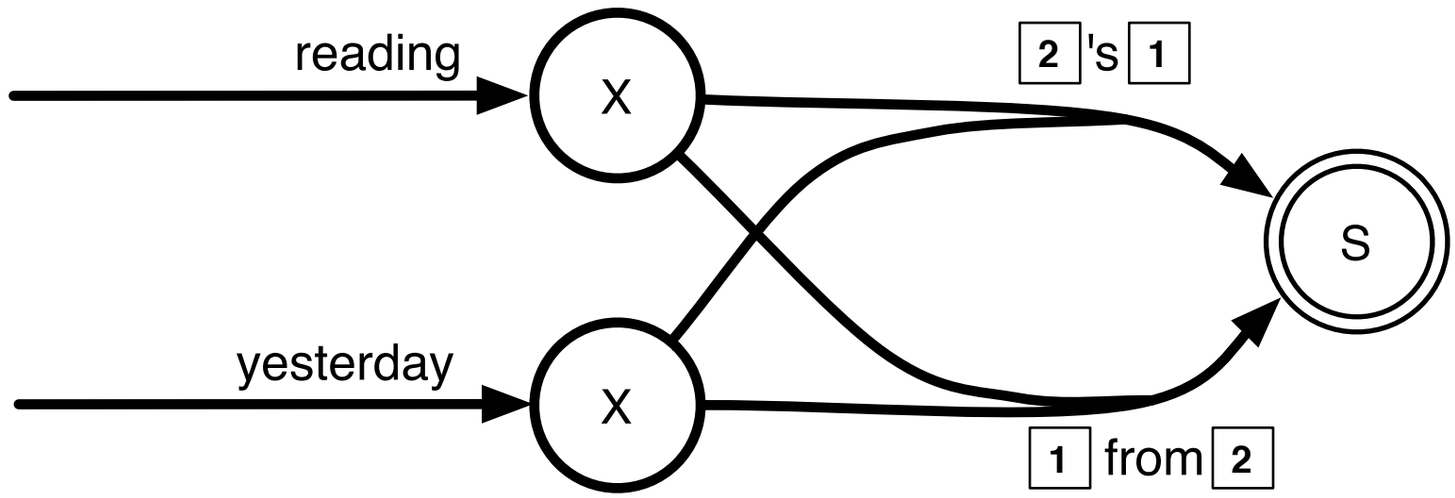
Substitution sites / variables / non-terminals

Hypergraph review



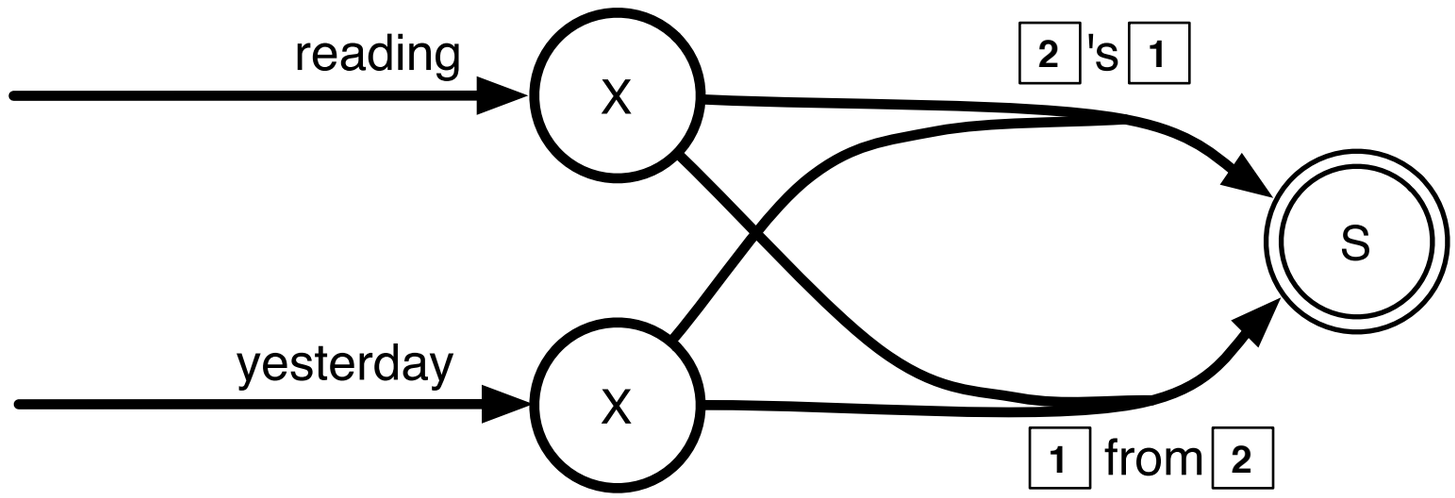
For LM integration, we ignore the source!

Hypergraph review



For LM integration, we ignore the source!

Hypergraph review



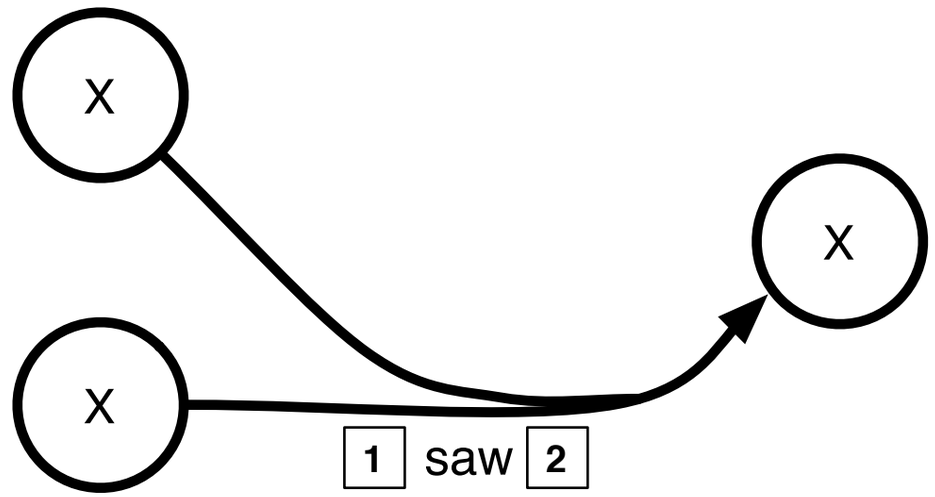
$$\{(\textit{yesterday 's reading}),\ (\textit{reading from yesterday})\}$$

How can we add the LM score to each string derived by the hypergraph?

LM Integration

- If LM features were purely local ...
  - "Unigram" model
- ... integration would be a breeze
  - Add an "LM feature" to every edge
- But, LM features are non-local!

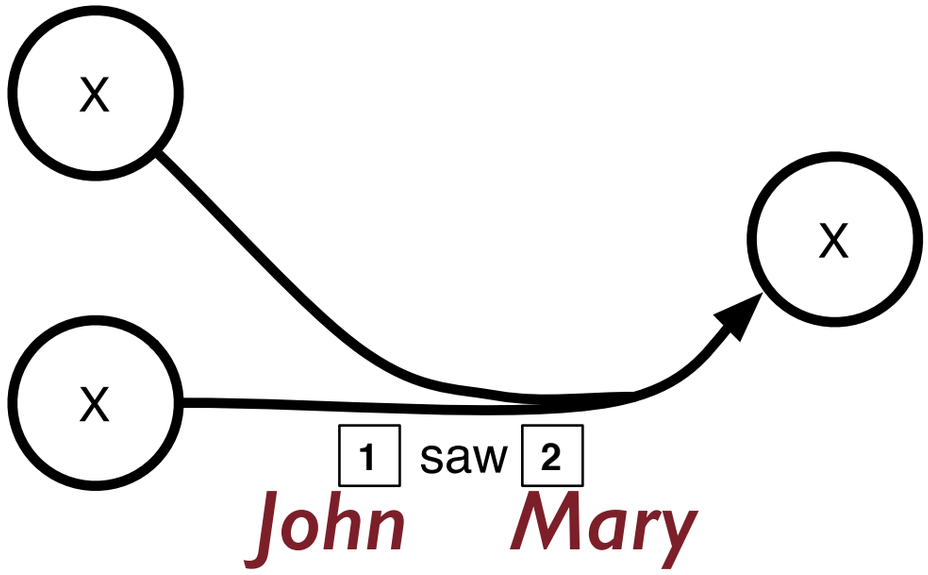
Why is it hard?



Two problems:

1. What is the content of the variables?

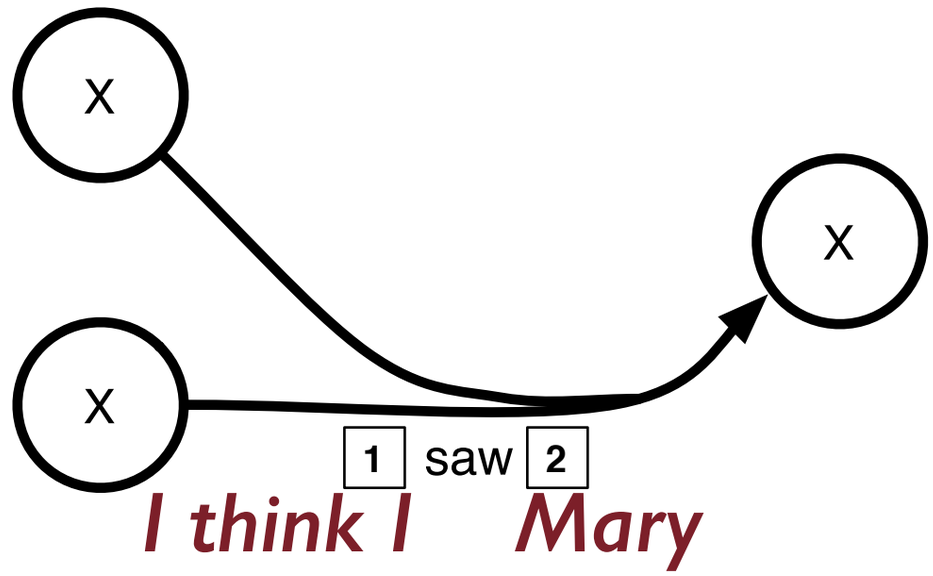
Why is it hard?



Two problems:

1. What is the content of the variables?

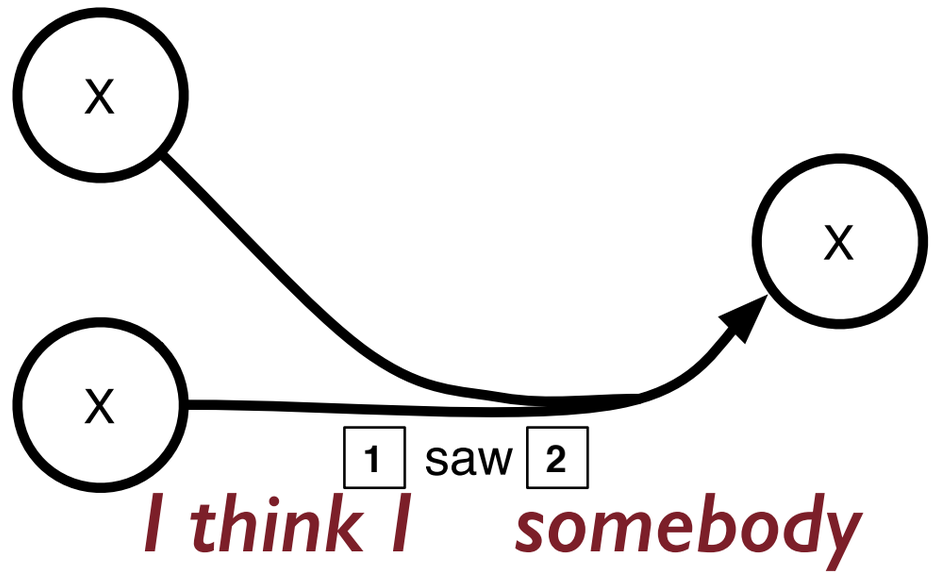
Why is it hard?



Two problems:

1. What is the content of the variables?

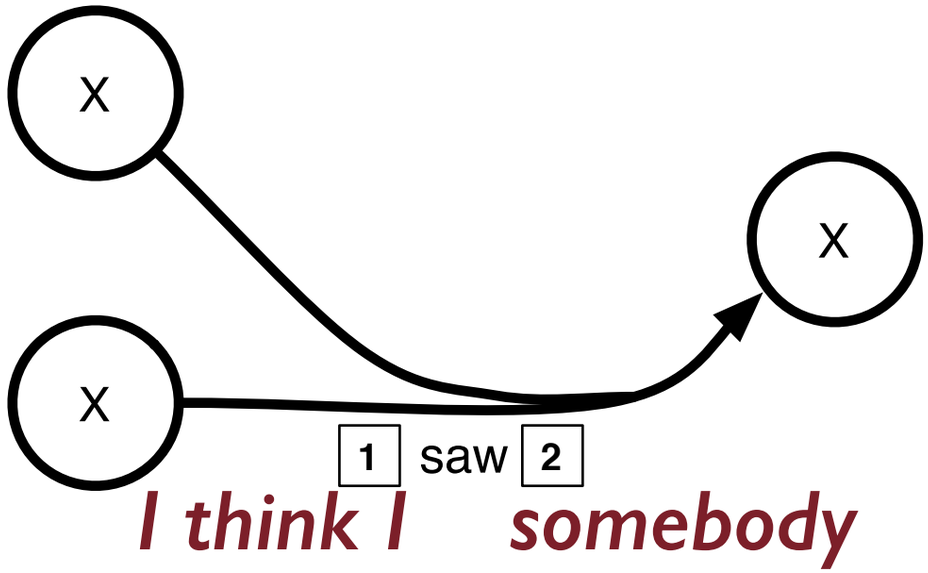
Why is it hard?



Two problems:

1. What is the content of the variables?

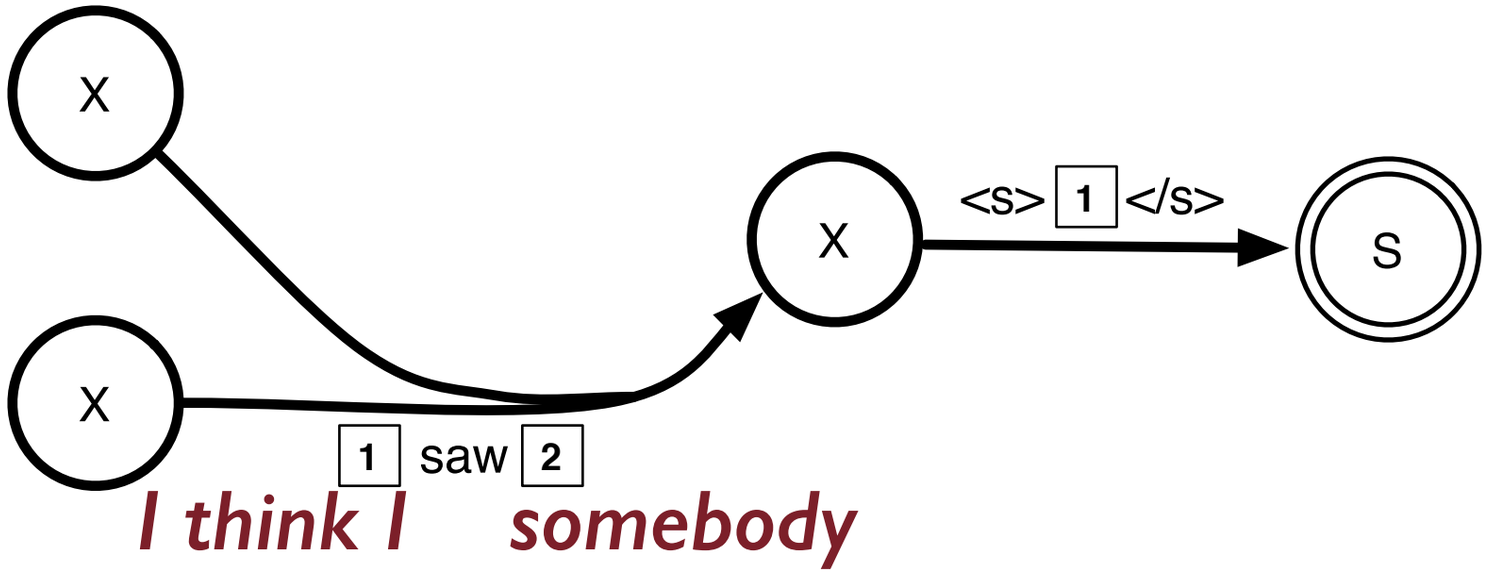
Why is it hard?



Two problems:

1. What is the content of the variables?
2. What will be the **left context** when this string is substituted somewhere?

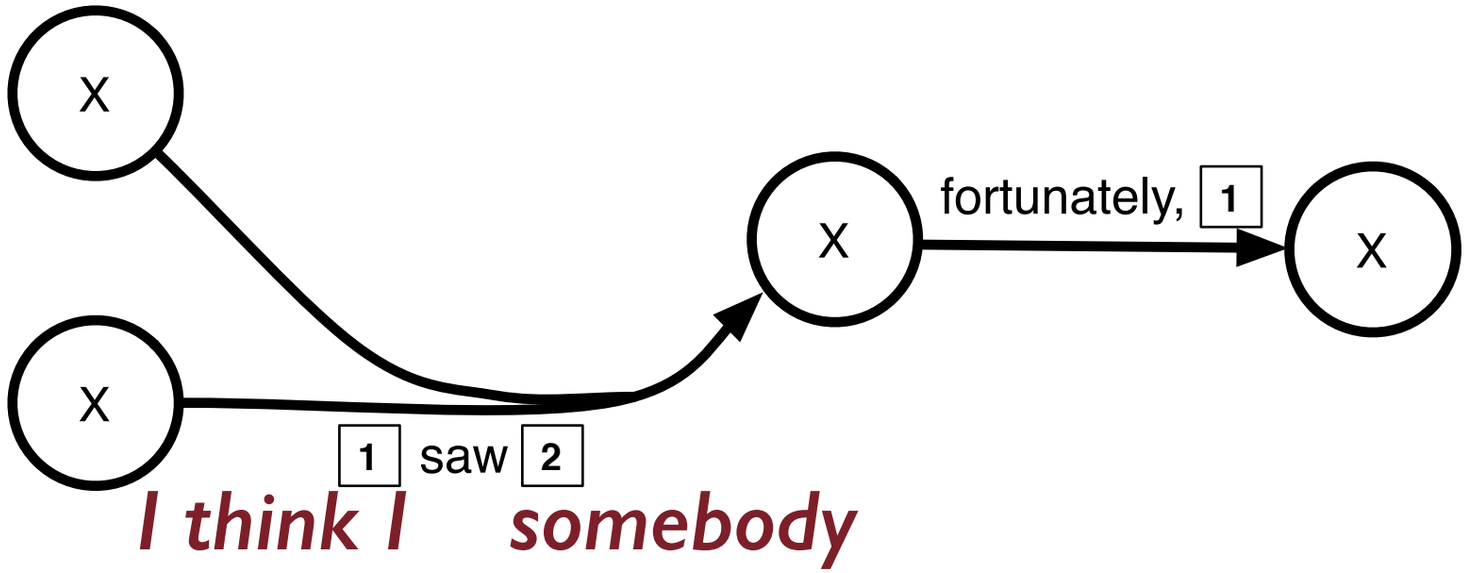
Why is it hard?



Two problems:

1. What is the content of the variables?
2. What will be the **left context** when this string is substituted somewhere?

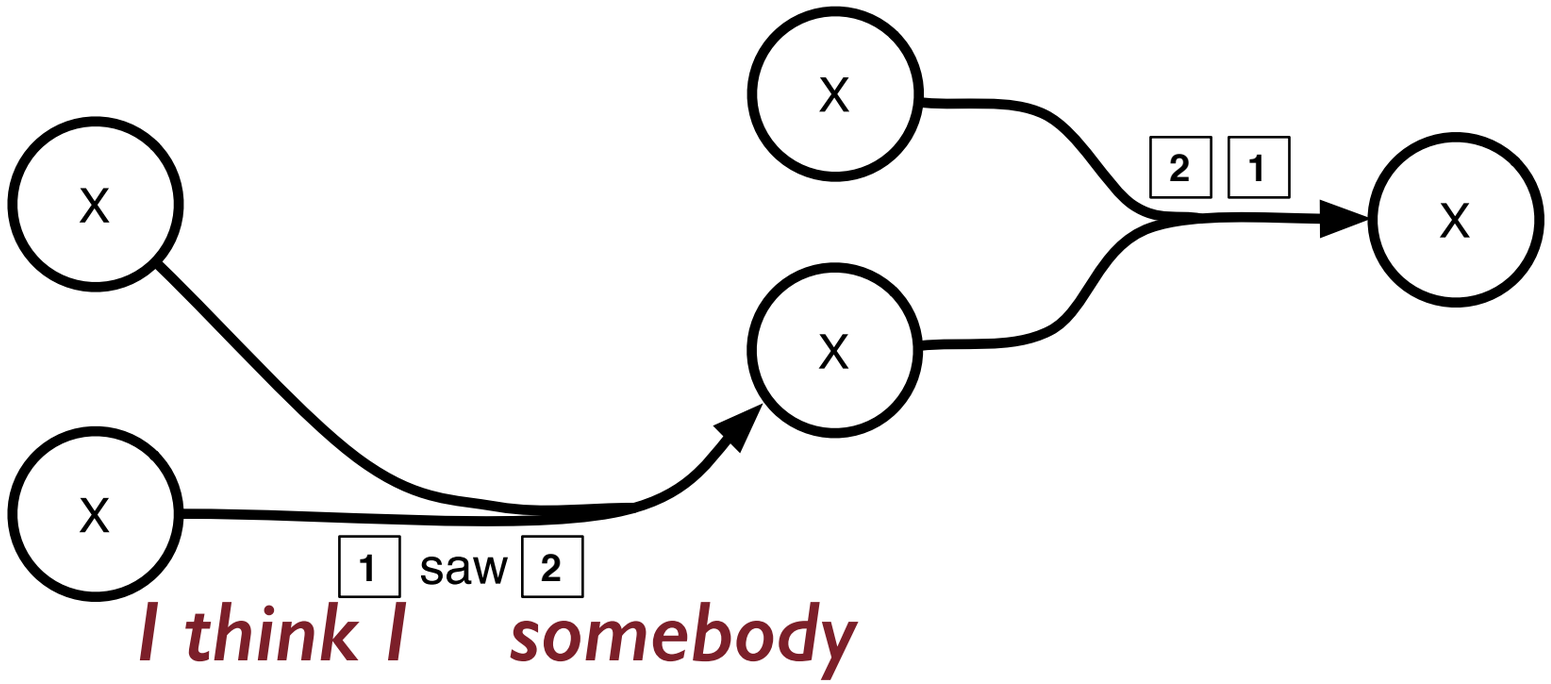
Why is it hard?



Two problems:

1. What is the content of the variables?
2. What will be the **left context** when this string is substituted somewhere?

Why is it hard?



Two problems:

1. What is the content of the variables?
2. What will be the **left context** when this string is substituted somewhere?

Naive solution

- Extract the all (k-best?) translations from the translation model
- Score them with an LM
- What's the problem with this?

Outline of DP solution

- Use n-order Markov assumption to help us
  - In an n-gram LM, words more than n words away will not affect the local (conditional) probability of a word in context
  - **This is not generally true, just the Markov assumption!**
- General approach
  - Restructure the hypergraph so that LM probabilities decompose along edges.
  - Solves both "problems"
    - we will not know the full value of variables, but we will know "enough".
    - defer scoring of left context until the context is established.

Hypergraph restructuring

- Note the following three facts:
  - If you know n or more consecutive words, the conditional probabilities of the nth, (n+1)th, ... words can be computed.
    - Therefore: add a feature weight to the edge for words.
  - (n-1) words of context to the **left** is enough to determine the probability of any word
    - Therefore: split nodes based on the (n-1) words on the **right** side of the span dominated by every node
  - (n-1) words on the **left** side of a span cannot be scored with certainty because the context is not known
    - Therefore: split nodes based on the (n-1) words on the **left** side of the span dominated by every node

Hypergraph restructuring

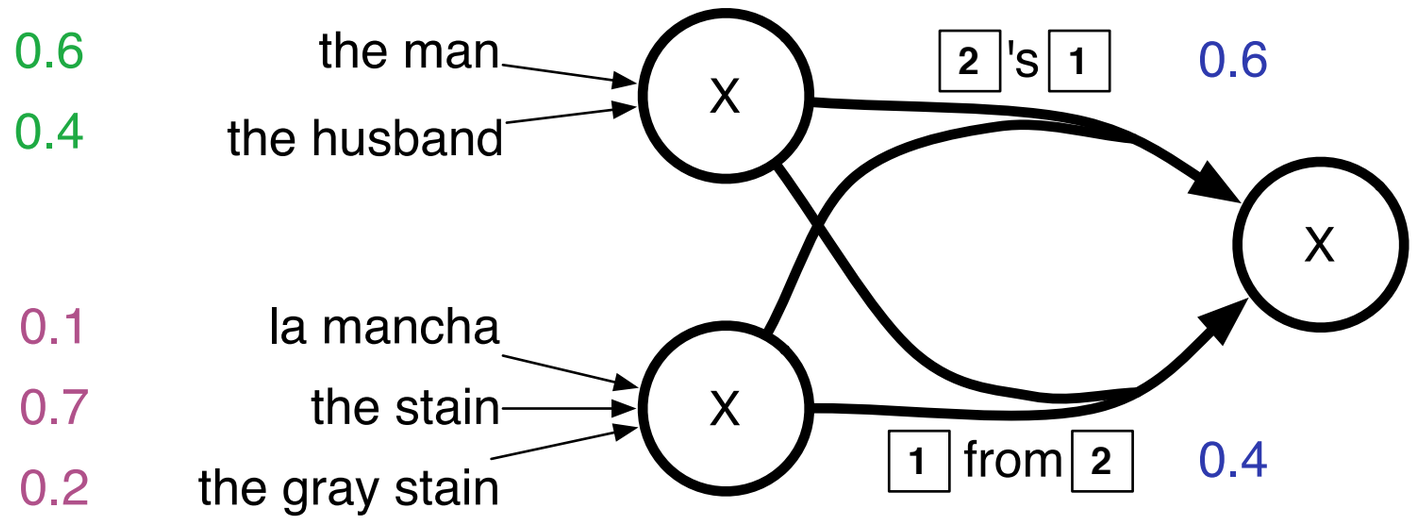
- Note the following three facts:
  - If you know n or more consecutive words, the conditional probabilities of the nth, $(n+1)$th, ... words can be computed.
    - [illegible]
  - (n [illegible] he pr [illegible]
    - [illegible] **ight** side of the span dominated by every node
  - $(n\text{-}1)$ words on the **left** side of a span cannot be scored with certainty because the context is not known
    - Therefore: split nodes based on the $(n\text{-}1)$ words on the **left** side of the span dominated by every node

Split nodes by the (n-1) words on both sides of the convergent edges.

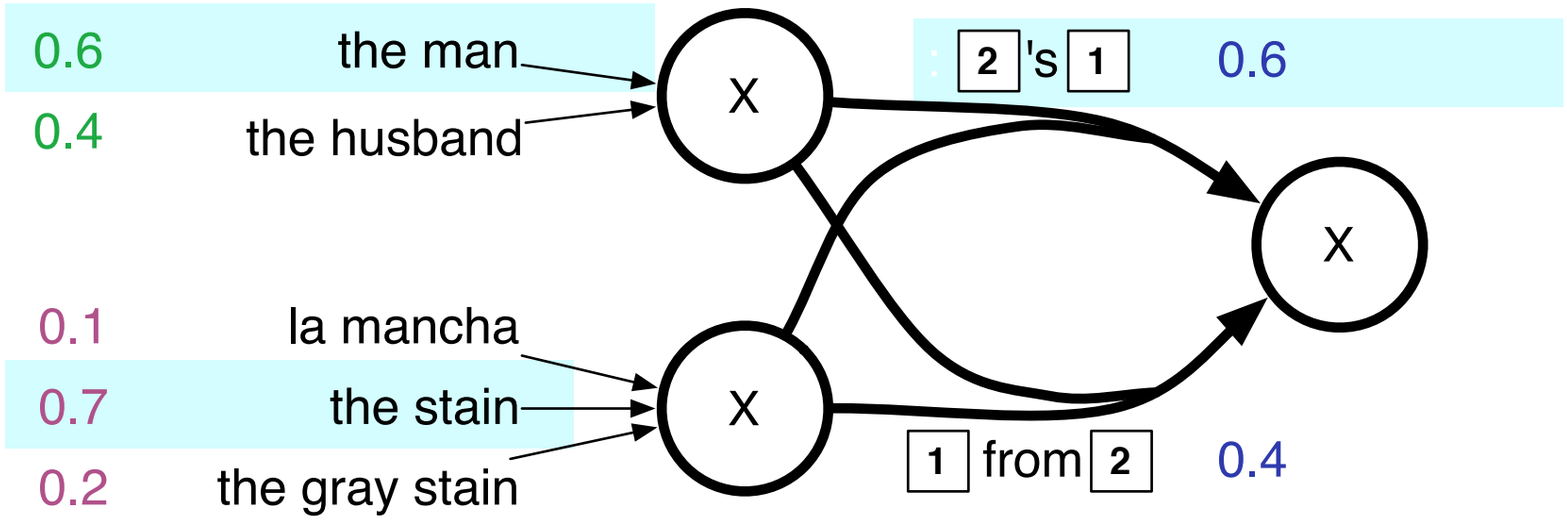
Hypergraph restructuring

- Algorithm ("cube intersection"):
  - For each node v (proceeding in **topological order** through the nodes)
    - For each edge e with head-node v, compute the (n-1) words on the left and right; call this q_e
      - Do this by substituting the (n-1)x2 word string from the tail node corresponding to the substitution variable
      - If node vq_e does not exist, create it, duplicating all outgoing edges from v so that they also proceed from vq_e
      - Disconnect e from v and attach it to vq_e
  - Delete v

Hypergraph restructuring



Hypergraph restructuring

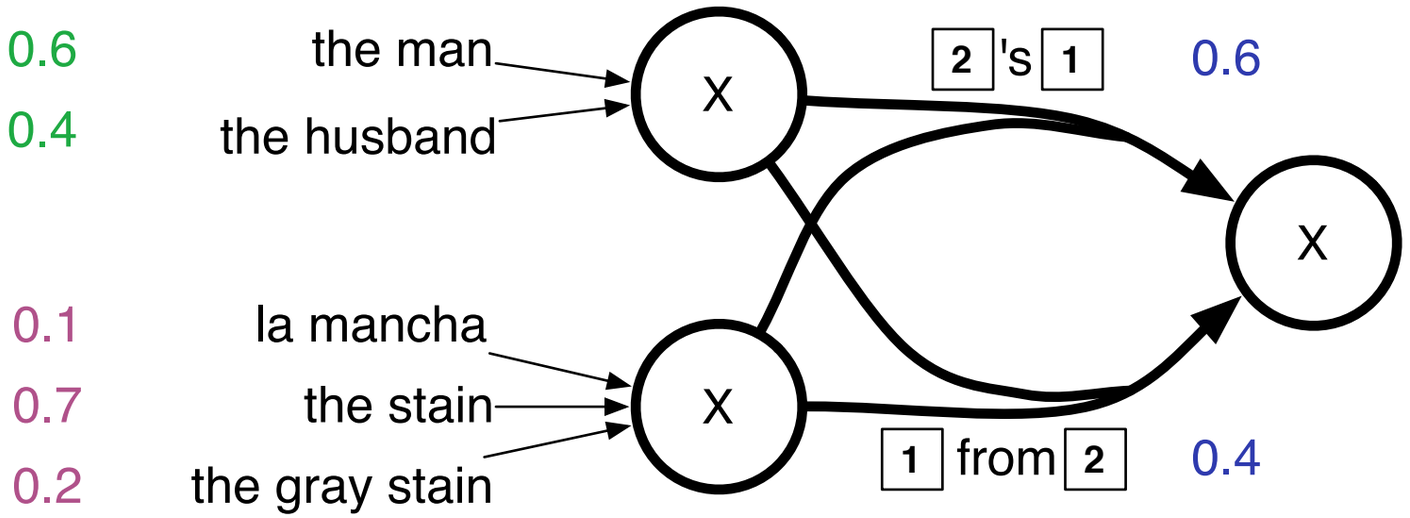


-LM Viterbi:

the stain's the man

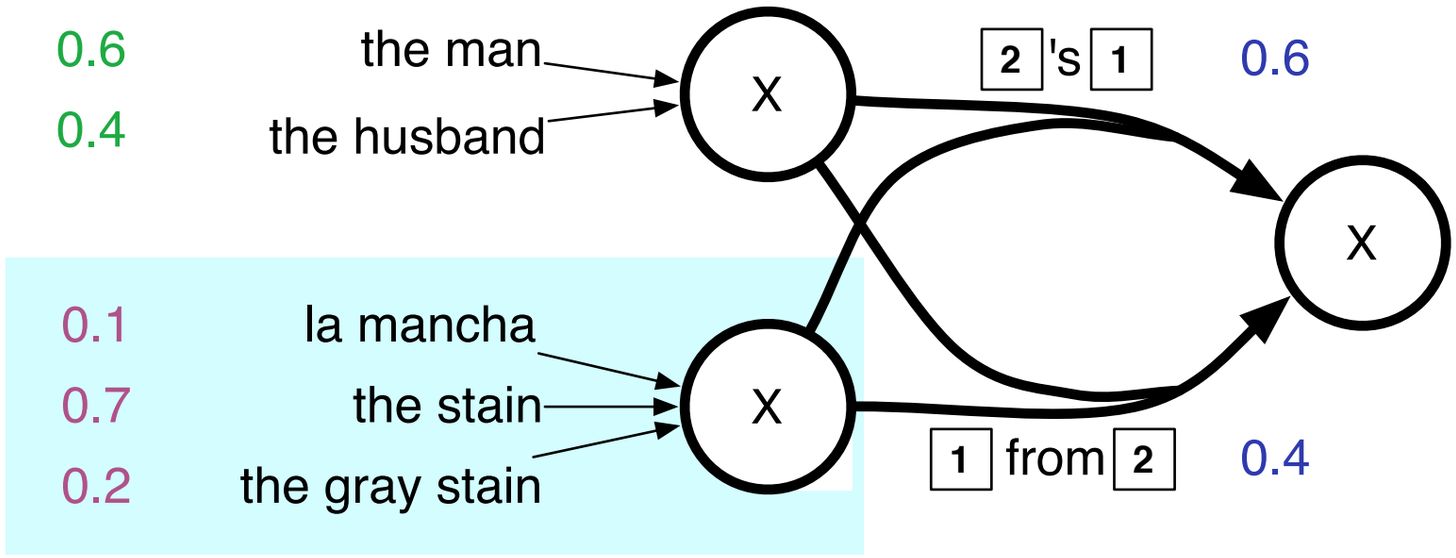
Hypergraph restructuring

Let's add a bi-gram language model!

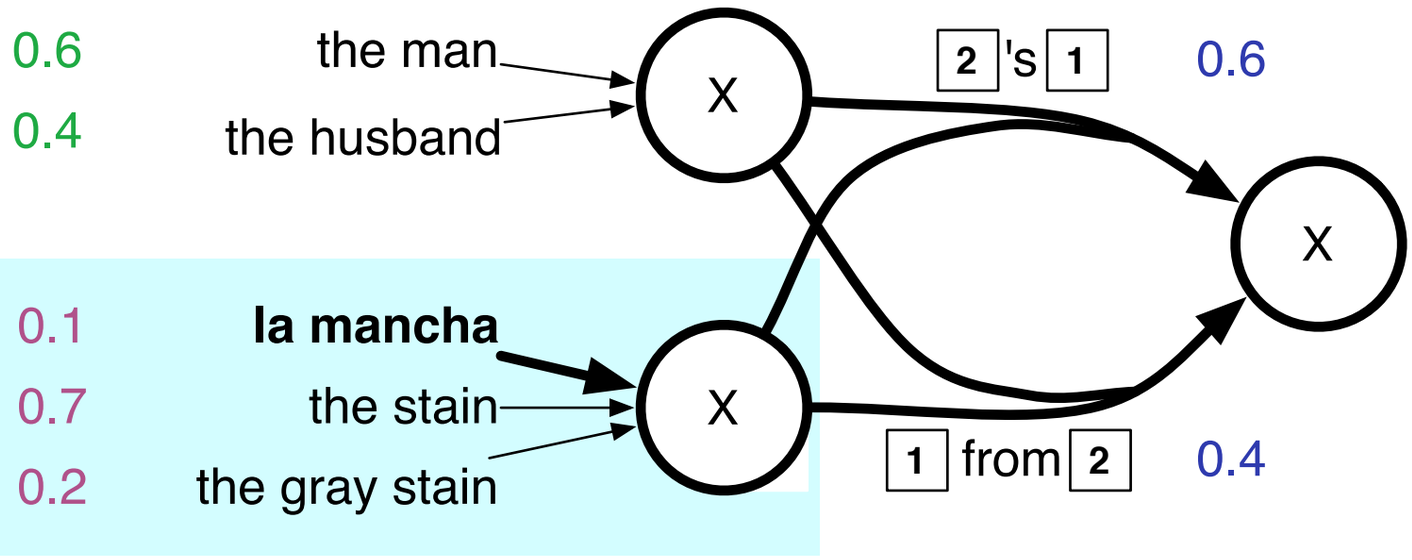


Hypergraph restructuring

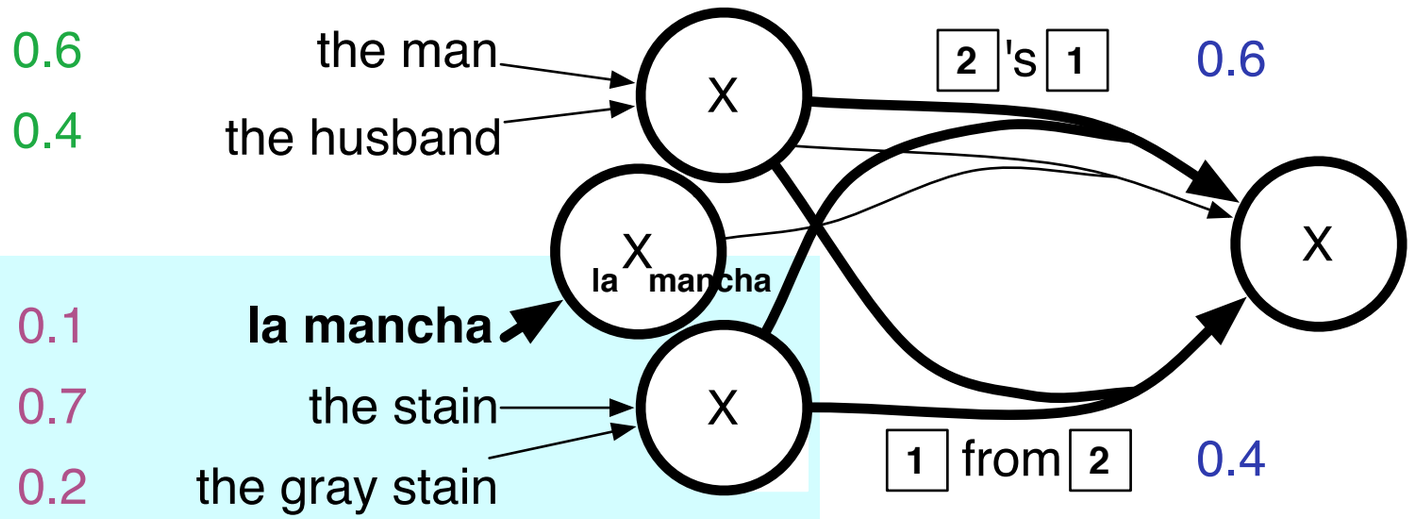
Let's add a bi-gram language model!



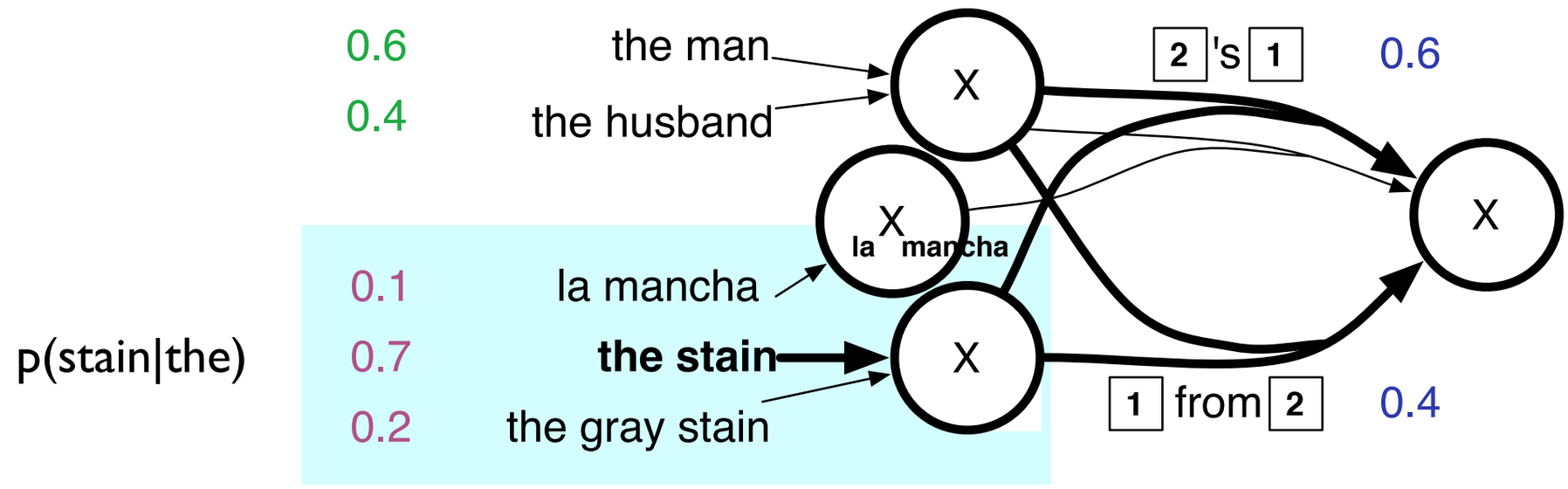
Hypergraph restructuring



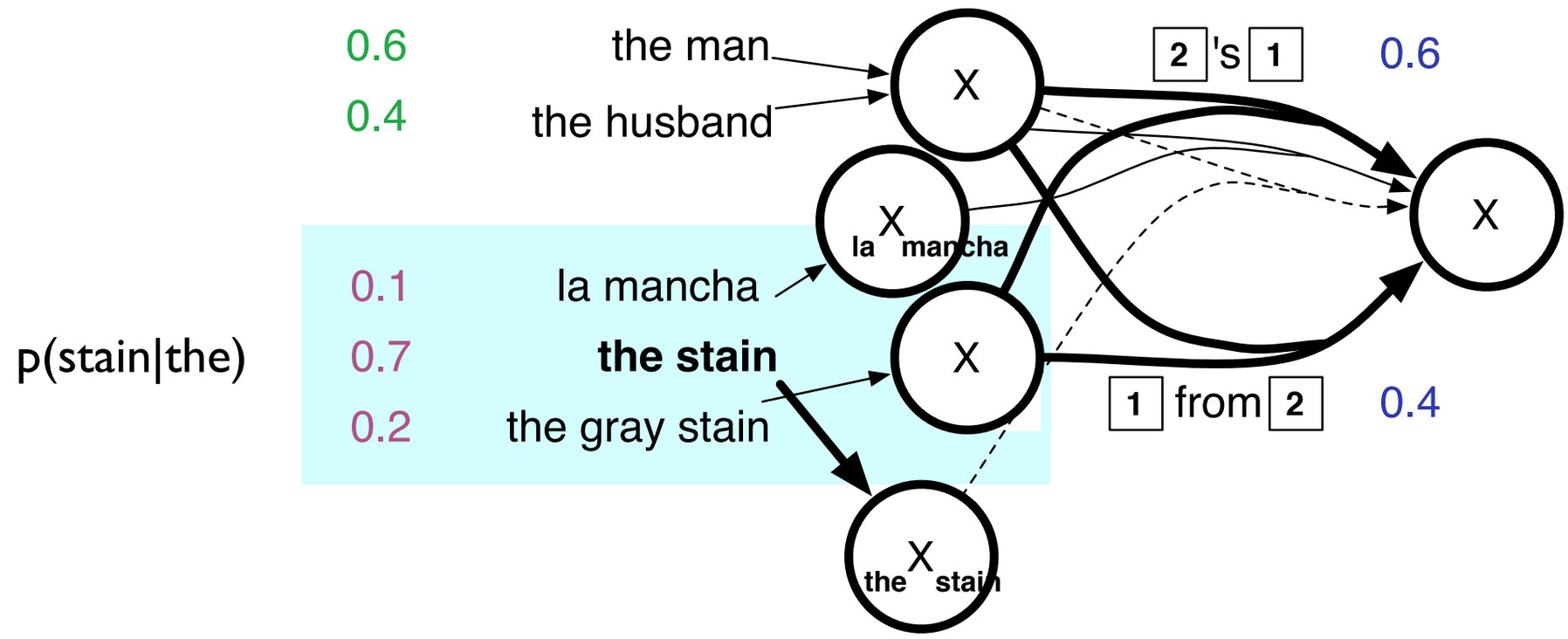
Hypergraph restructuring



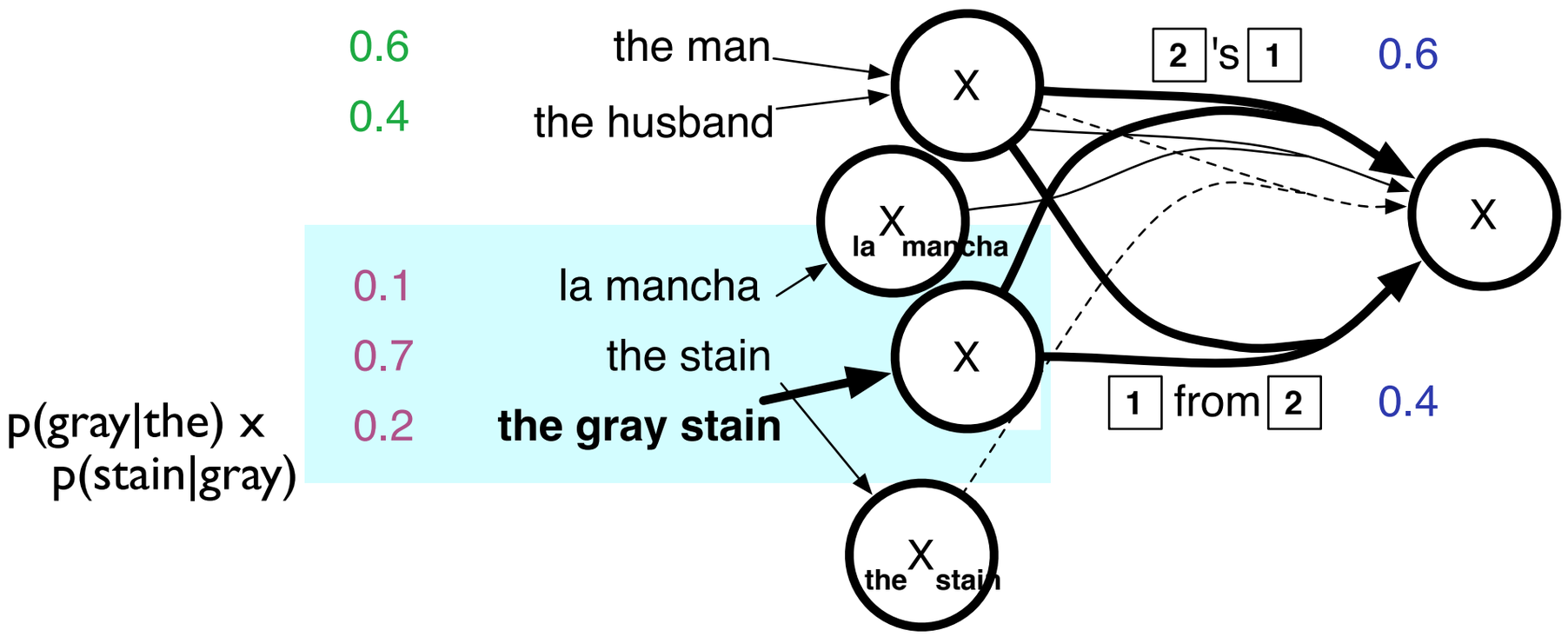
Hypergraph restructuring



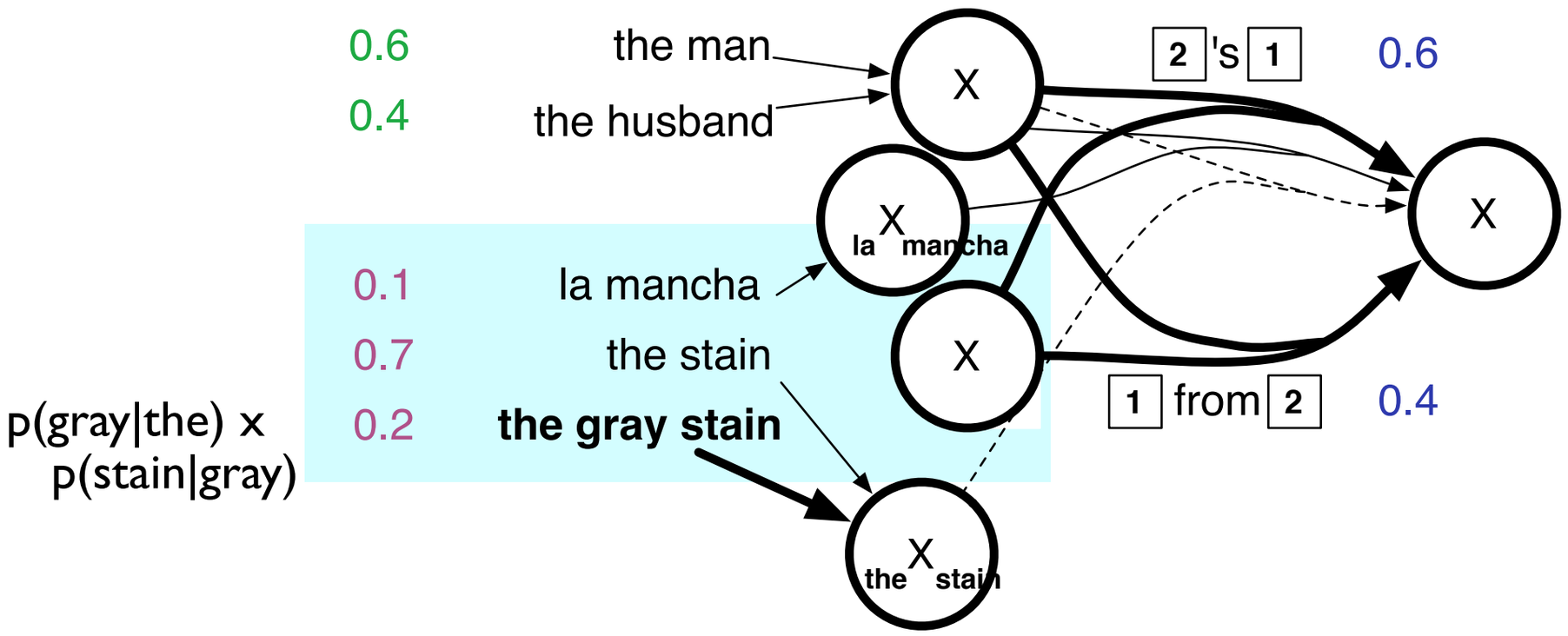
Hypergraph restructuring



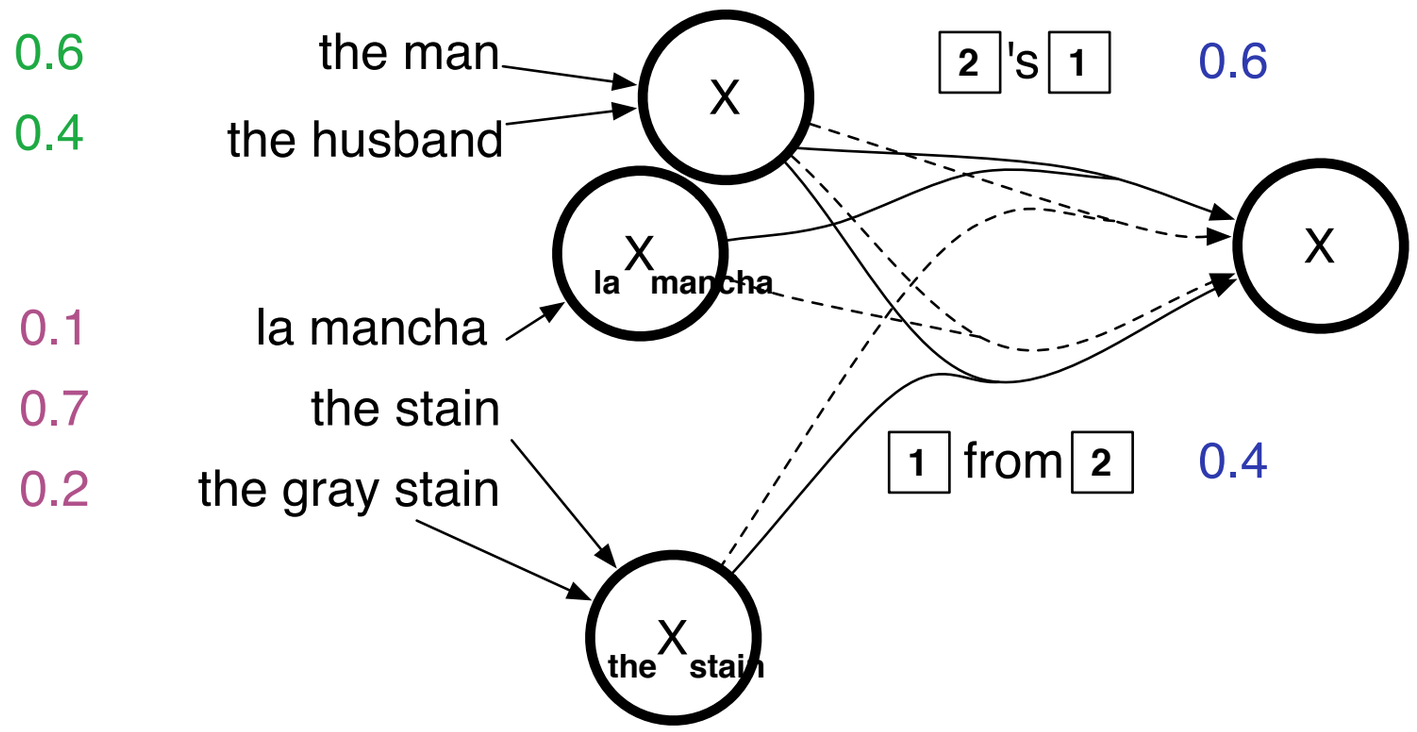
Hypergraph restructuring



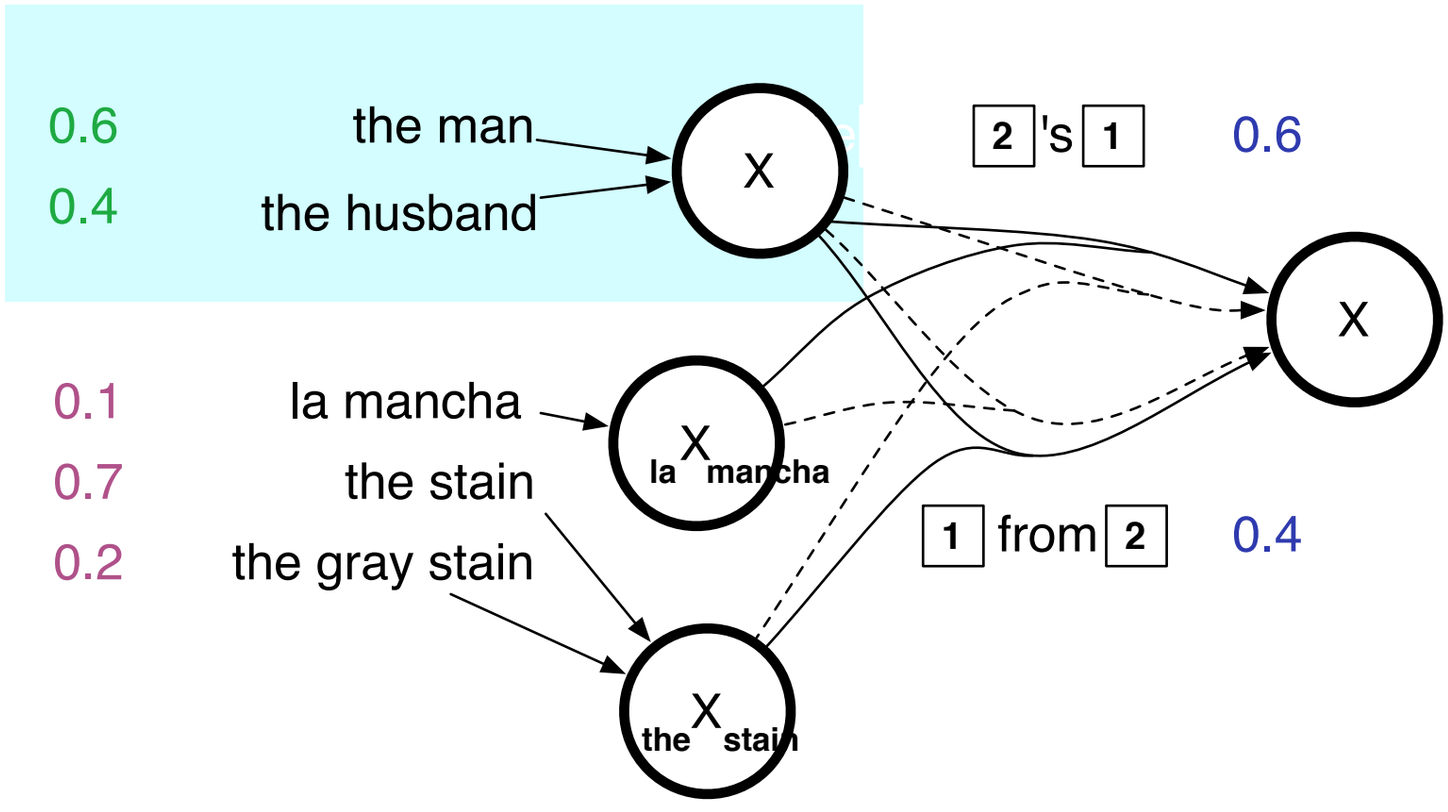
Hypergraph restructuring



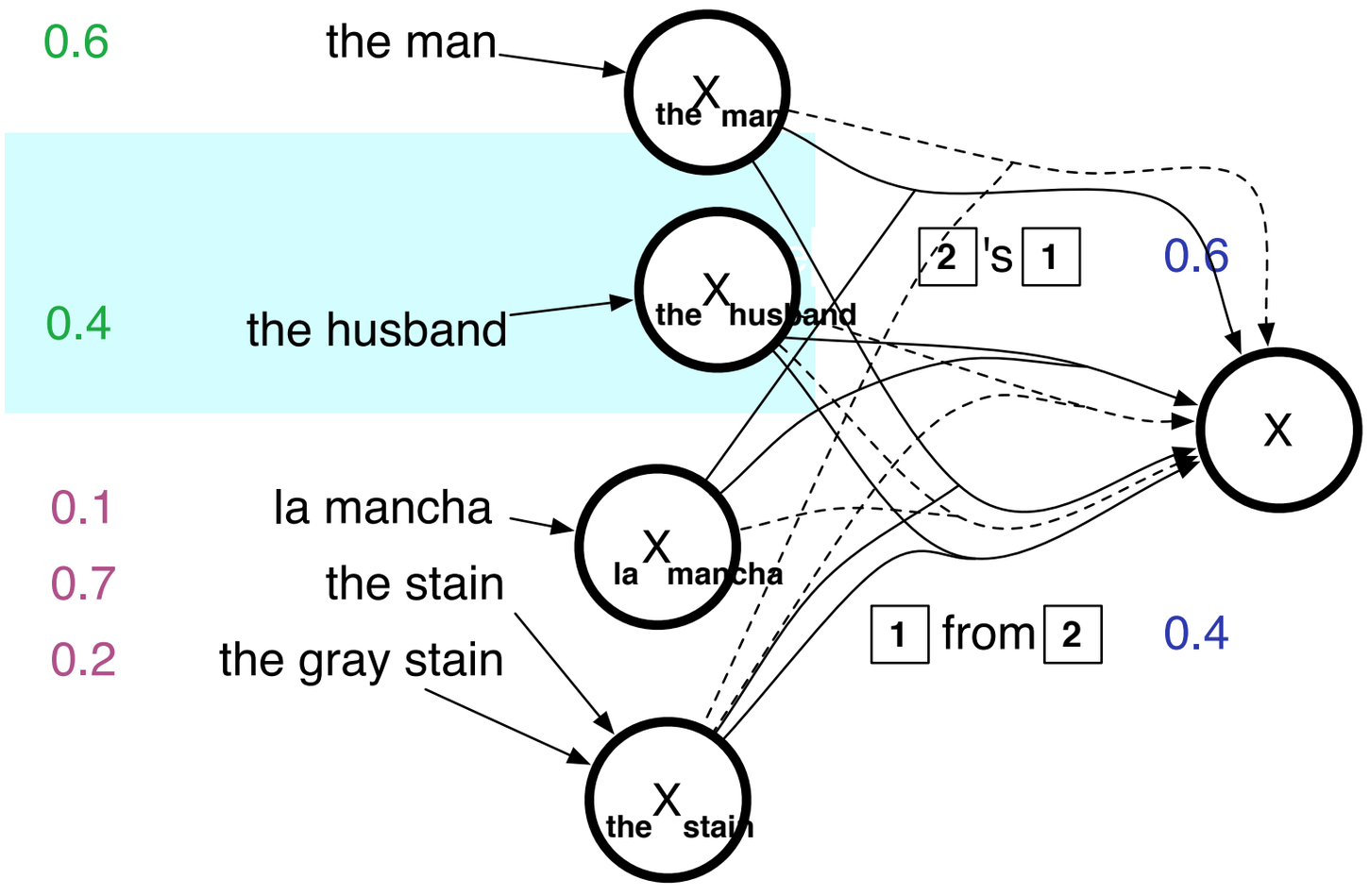
Hypergraph restructuring



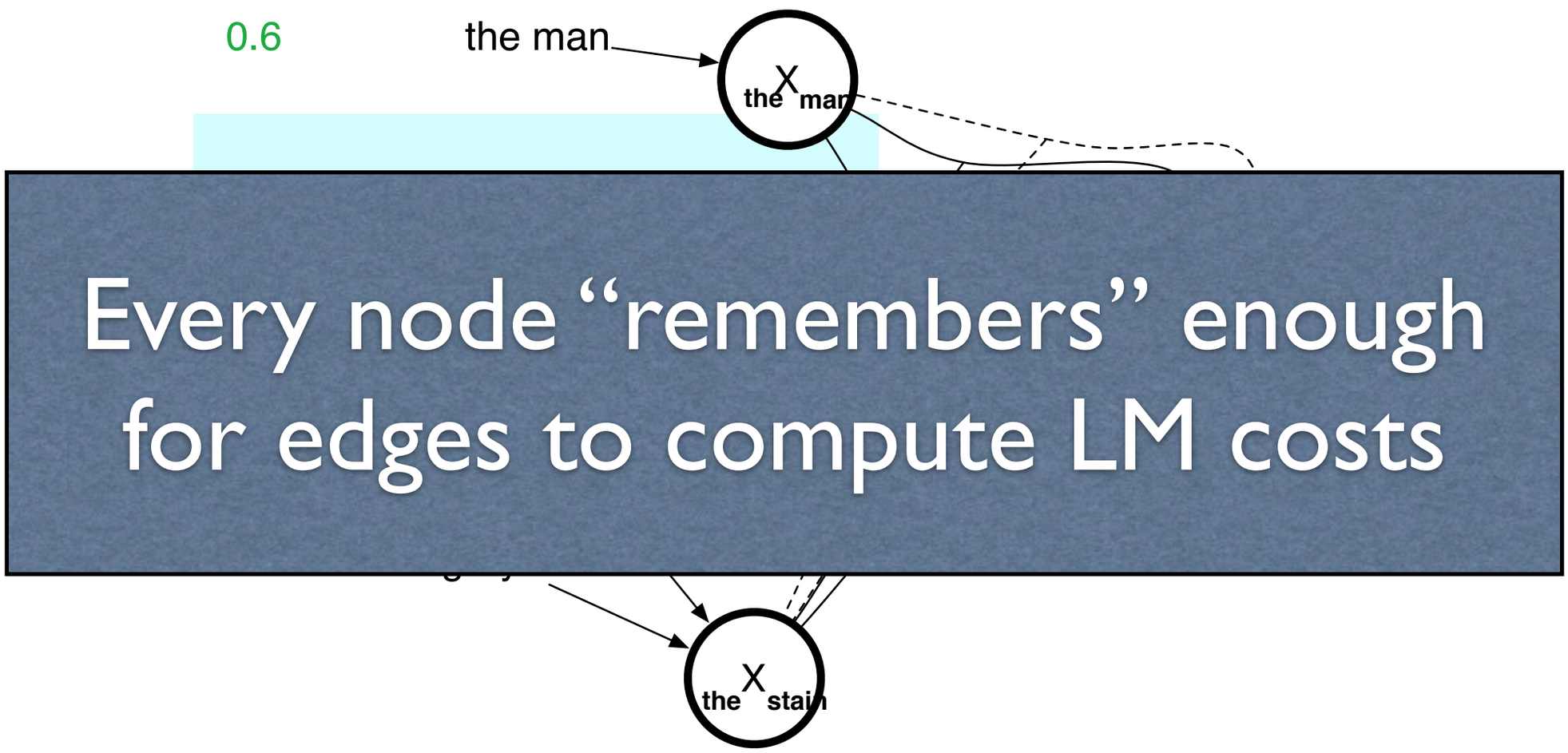
Hypergraph restructuring



Hypergraph restructuring



Hypergraph restructuring



Every node "remembers" enough for edges to compute LM costs

Complexity

- What is the run-time of this algorithm?

Complexity

- What is the run-time of this algorithm?

$$O(|V||E||\Sigma|^{2(n-1)})$$

Going to longer n-grams is exponentially expensive!

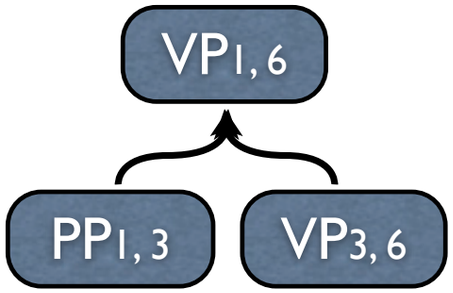
Cube pruning

- Expanding every node like this exhaustively is impractical
  - Polynomial time, but really, really big!
- Cube pruning: minor tweak on the above algorithm
  - Compute the k-best expansions at each node
  - Use an **estimate** (usually a unigram probability) of the unscored left-edge to rank the nodes

Cube pruning

- Why "cube" pruning?
  - Cube-pruning only involves a "cube" when arity-2 rules are used!
  - More appropriately called "square" pruning with arity-1
  - Or "hypercube" pruning with arity > 2!

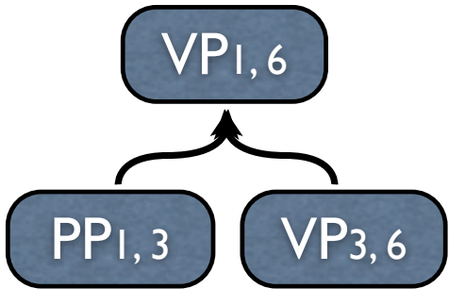
Cube Pruning



monotonic grid?

	$(\text{PP}^{\text{with} \star \text{Sharon}}_{1,3})$	$(\text{PP}^{\text{along} \star \text{Sharon}}_{1,3})$	$(\text{PP}^{\text{with} \star \text{Shalong}}_{1,3})$
	1.0	3.0	8.0
$(\text{VP}^{\text{held} \star \text{meeting}}_{3,6})$ 1.0	2.0	4.0	9.0
$(\text{VP}^{\text{held} \star \text{talk}}_{3,6})$ 1.1	2.1	4.1	9.1
$(\text{VP}^{\text{hold} \star \text{conference}}_{3,6})$ 3.5	4.5	6.5	11.5

Cube Pruning



non-monotonic grid
due to LM combo costs

$(\text{PP}^{\text{with} \star \text{Sharon}}_{1,3})$ $(\text{PP}^{\text{along} \star \text{Sharon}}_{1,3})$ $(\text{PP}^{\text{with} \star \text{Shalong}}_{1,3})$

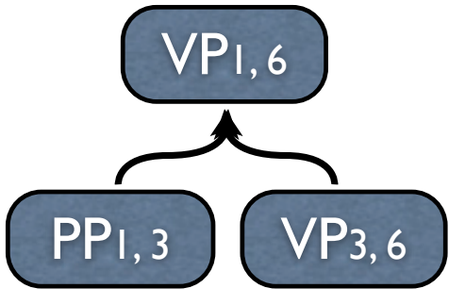
$(\text{VP}^{\text{held} \star \text{meeting}}_{3,6})$

$(\text{VP}^{\text{held} \star \text{talk}}_{3,6})$

$(\text{VP}^{\text{hold} \star \text{conference}}_{3,6})$

	1.0	3.0	8.0
1.0	2.0 + 0.5	4.0 + 5.0	9.0 + 0.5
1.1	2.1 + 0.3	4.1 + 5.4	9.1 + 0.3
3.5	4.5 + 0.6	6.5 +10.5	11.5 + 0.6

Cube Pruning



bigram (meeting, with)

$(\text{PP}^{\text{with}\star\text{Sharon}}_{1,3})$ $(\text{PP}^{\text{along}\star\text{Sharon}}_{1,3})$ $(\text{PP}^{\text{with}\star\text{Shalong}}_{1,3})$

non-monotonic grid
due to LM combo costs

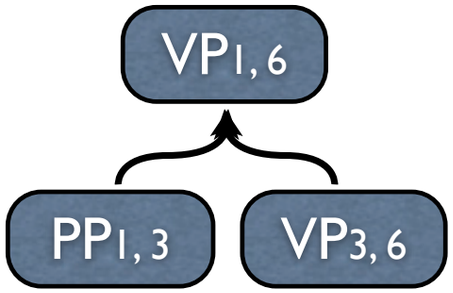
$(\text{VP}^{\text{held}\star\text{meeting}}_{3,6})$

$(\text{VP}^{\text{held}\star\text{talk}}_{3,6})$

$(\text{VP}^{\text{hold}\star\text{conference}}_{3,6})$

	1.0	3.0	8.0
1.0	2.0 + 0.5	4.0 + 5.0	9.0 + 0.5
1.1	2.1 + 0.3	4.1 + 5.4	9.1 + 0.3
3.5	4.5 + 0.6	6.5 + 10.5	11.5 + 0.6

Cube Pruning



non-monotonic grid
due to LM combo costs

	$(\text{PP}^{\text{with} \star \text{Sharon}}_{1,3})$	$(\text{PP}^{\text{along} \star \text{Sharon}}_{1,3})$	$(\text{PP}^{\text{with} \star \text{Shalong}}_{1,3})$
	1.0	3.0	8.0
$(\text{VP}^{\text{held} \star \text{meeting}}_{3,6})$ 1.0	2.5	9.0	9.5
$(\text{VP}^{\text{held} \star \text{talk}}_{3,6})$ 1.1	2.4	9.5	9.4
$(\text{VP}^{\text{hold} \star \text{conference}}_{3,6})$ 3.5	5.1	17.0	12.1

Cube Pruning

k-best parsing
(Huang and Chiang, 2005)

- a priority queue of candidates
- extract the best candidate

$(\text{PP}_{1,3}^{\text{with} \star \text{Sharon}})$ $(\text{PP}_{1,3}^{\text{along} \star \text{Sharon}})$ $(\text{PP}_{1,3}^{\text{with} \star \text{Shalong}})$

$(\text{VP}_{3,6}^{\text{held} \star \text{meeting}})$

$(\text{VP}_{3,6}^{\text{held} \star \text{talk}})$

$(\text{VP}_{3,6}^{\text{hold} \star \text{conference}})$

	1.0	3.0	8.0
1.0	2.5	9.0	9.5
1.1	2.4	9.5	9.4
3.5	5.1	17.0	12.1

Cube Pruning

k-best parsing
(Huang and Chiang, 2005)

- a priority queue of candidates
- extract the best candidate
- push the two successors

$(\text{PP}^{\text{with} \star \text{Sharon}}_{1,3})$ $(\text{PP}^{\text{along} \star \text{Sharon}}_{1,3})$ $(\text{PP}^{\text{with} \star \text{Shalong}}_{1,3})$

$(\text{VP}^{\text{held} \star \text{meeting}}_{3,6})$

$(\text{VP}^{\text{held} \star \text{talk}}_{3,6})$

$(\text{VP}^{\text{hold} \star \text{conference}}_{3,6})$

	1.0	3.0	8.0
1.0	2.5	9.0	9.5
1.1	2.4	9.5	9.4
3.5	5.1	17.0	12.1

Cube Pruning

k-best parsing
(Huang and Chiang, 2005)

- a priority queue of candidates
- extract the best candidate
- push the two successors

$(\text{PP}_{1,3}^{\text{with} \star \text{Sharon}})$ $(\text{PP}_{1,3}^{\text{along} \star \text{Sharon}})$ $(\text{PP}_{1,3}^{\text{with} \star \text{Shalong}})$

$(\text{VP}_{3,6}^{\text{held} \star \text{meeting}})$

$(\text{VP}_{3,6}^{\text{held} \star \text{talk}})$

$(\text{VP}_{3,6}^{\text{hold} \star \text{conference}})$

	1.0	3.0	8.0
1.0	2.5	9.0	9.5
1.1	2.4	9.5	9.4
3.5	5.1	17.0	12.1

Cube pruning

- Widely used for phrase-based and syntax-based MT
- May be applied in conjunction with a bottom-up decoder, or as a second "rescoring" pass
  - Nodes may also be grouped together (for example, all nodes corresponding to a certain source span)
- Requirement for topological ordering means translation hypergraph may not have cycles