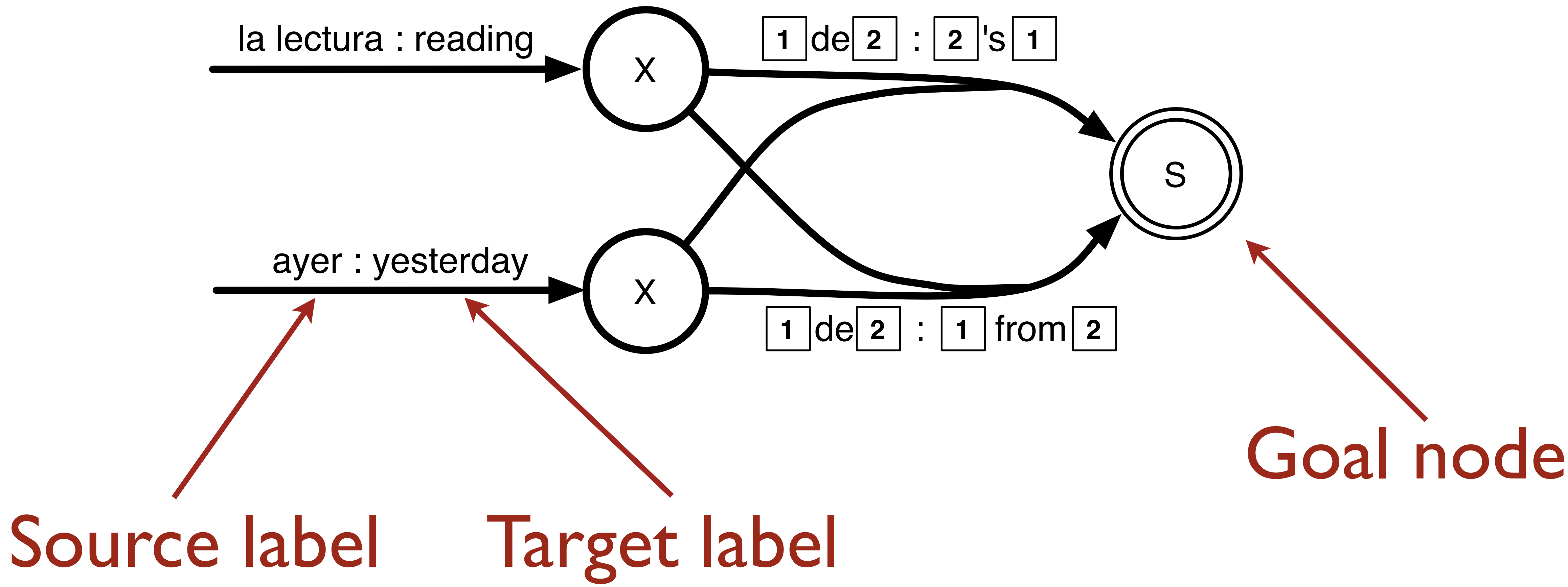
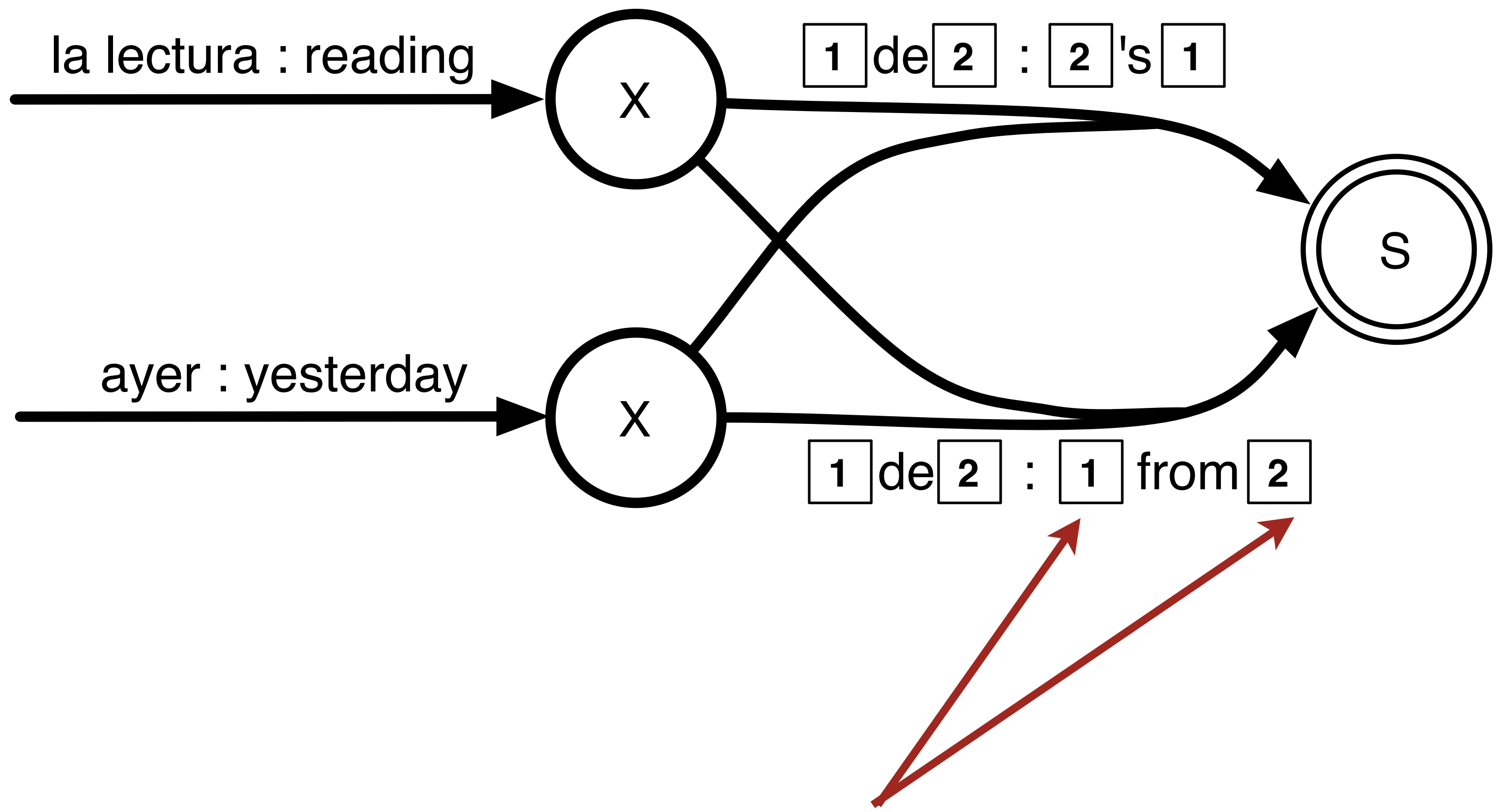
Using N-Gram LMs with SCFG TMs

March 20, 2013

Hypergraph review

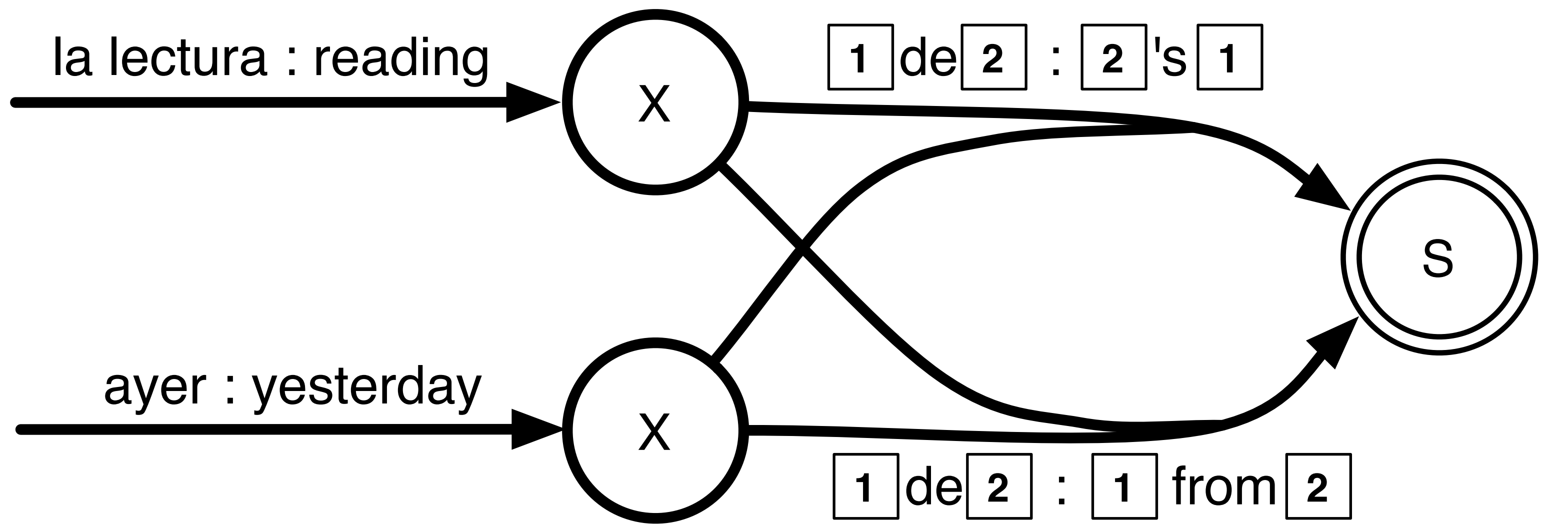


Hypergraph review



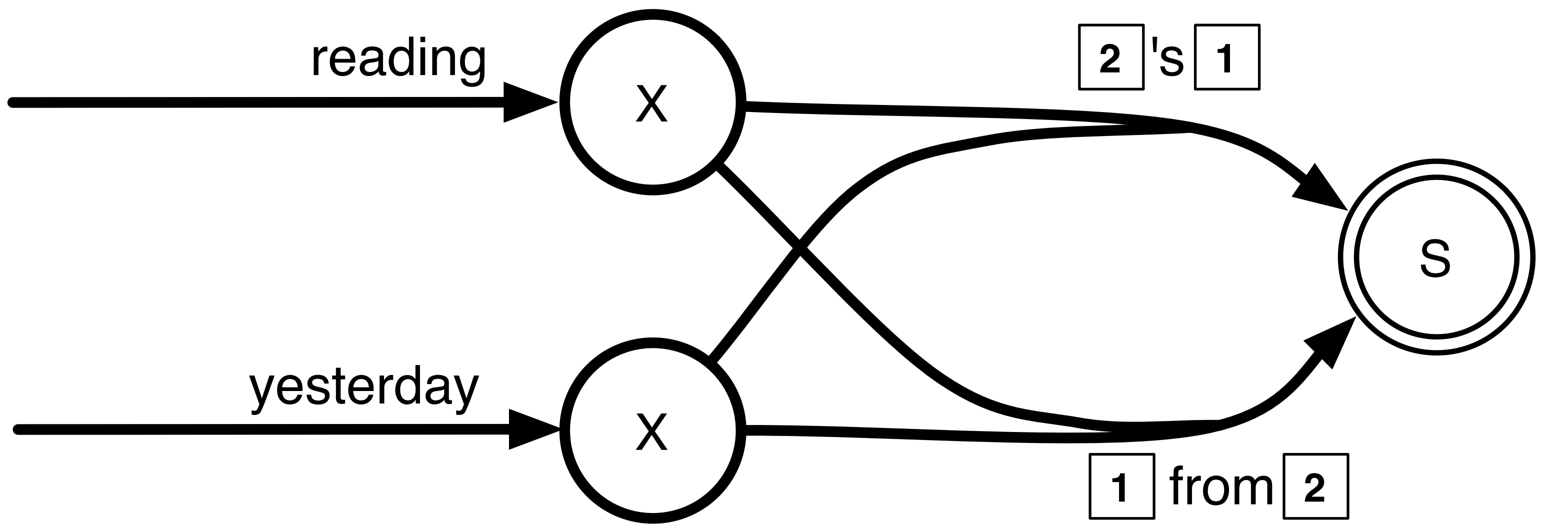
Substitution sites / variables / non-terminals

Hypergraph review



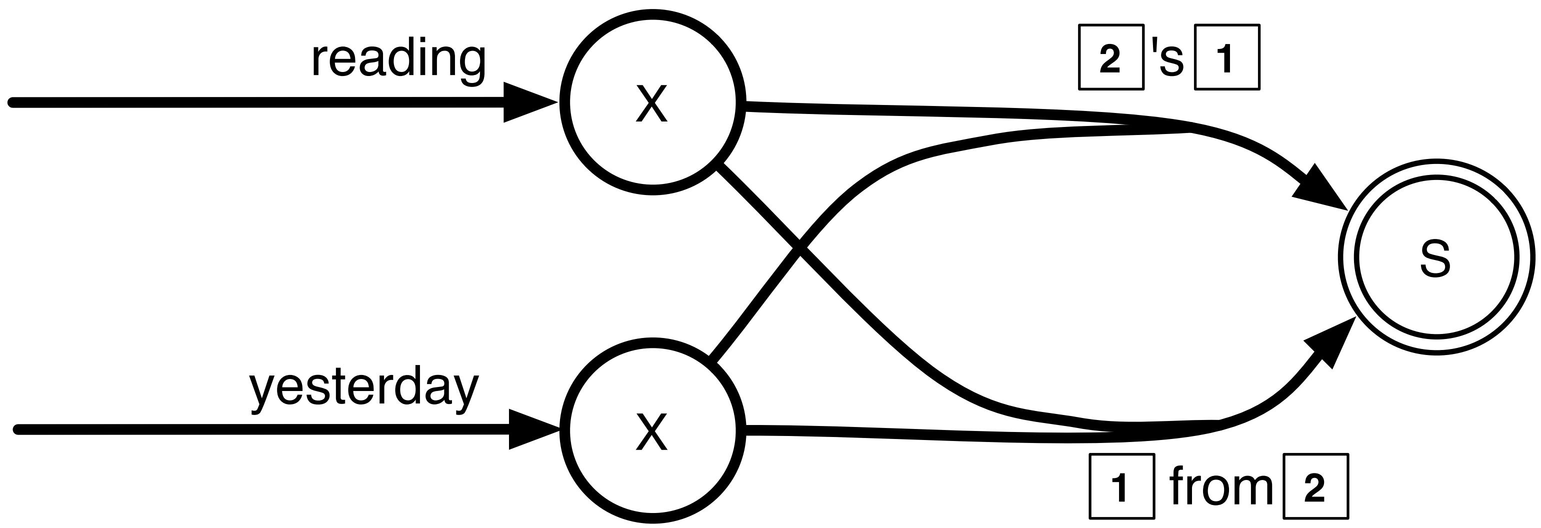
For LM integration, we ignore the source!

Hypergraph review



For LM integration, we ignore the source!

Hypergraph review



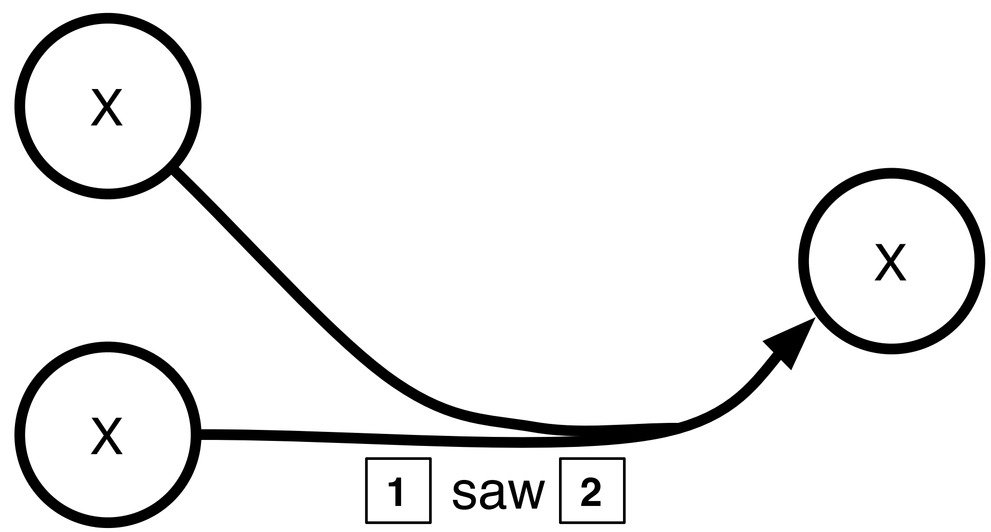
$\{$(*yesterday ’s reading*),
(*reading from yesterday*)$\}$

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

LM Integration

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

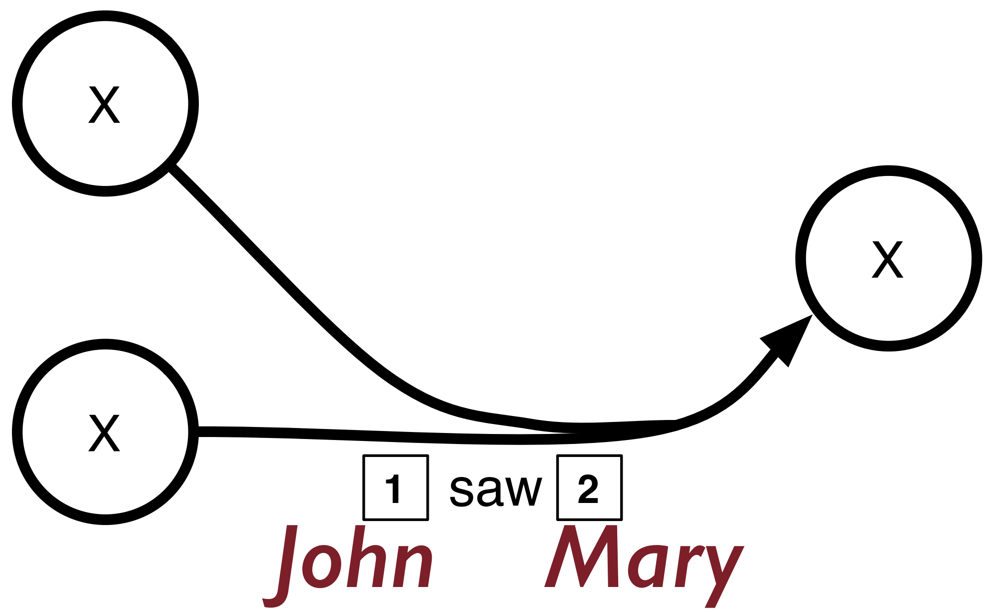
Why is it hard?



Two problems:

1. What is the content of the variables?

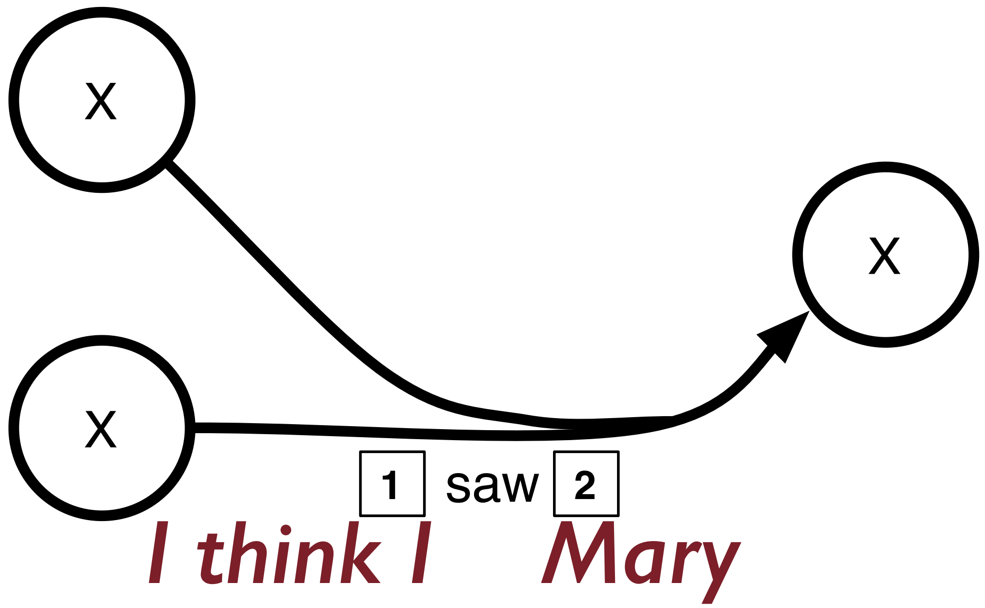
Why is it hard?



Two problems:

1. What is the content of the variables?

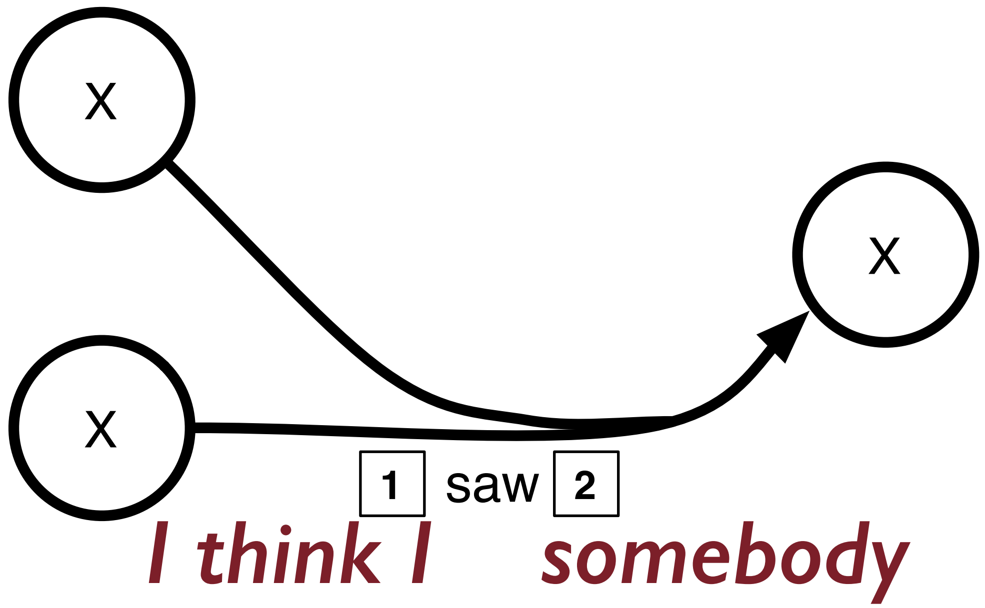
Why is it hard?



Two problems:

1. What is the content of the variables?

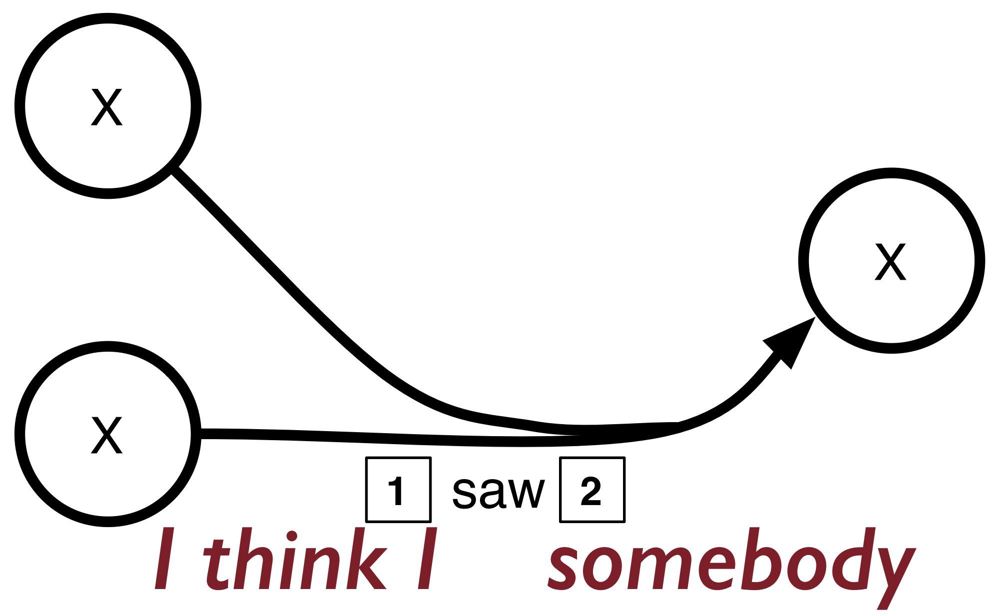
Why is it hard?



Two problems:

1. What is the content of the variables?

Why is it hard?

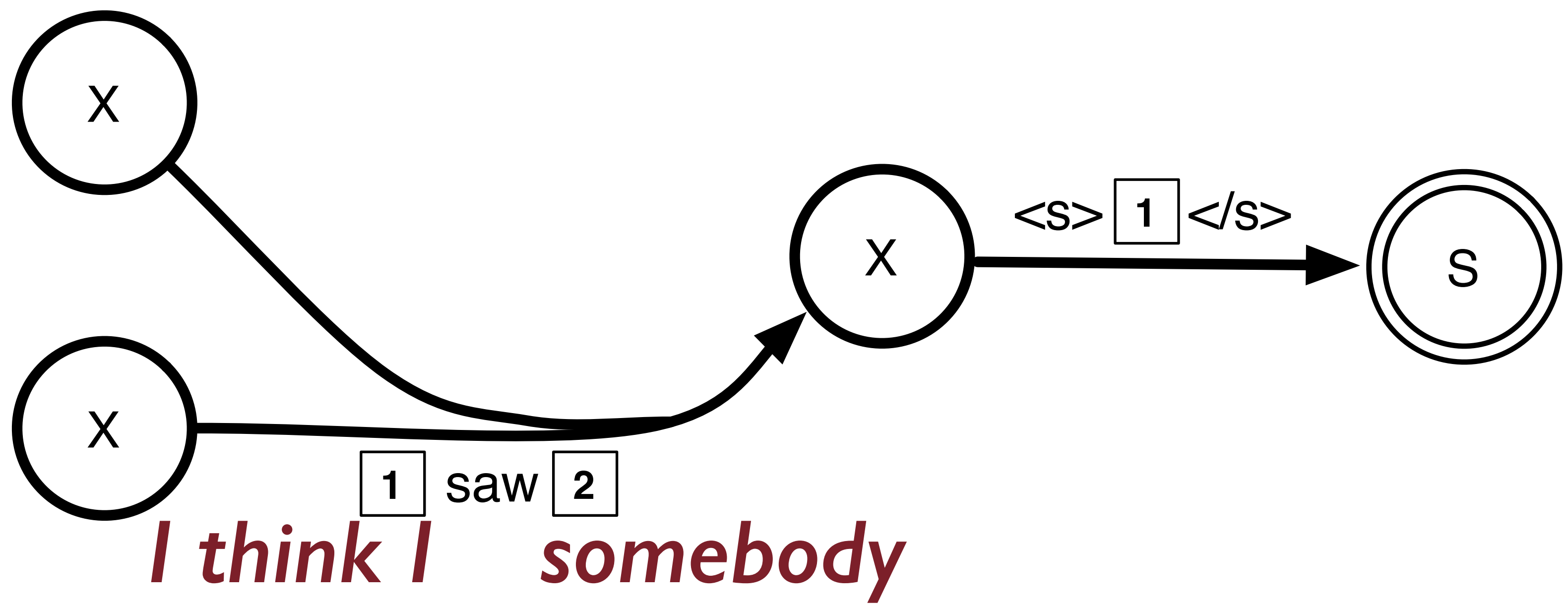


Two problems:

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?

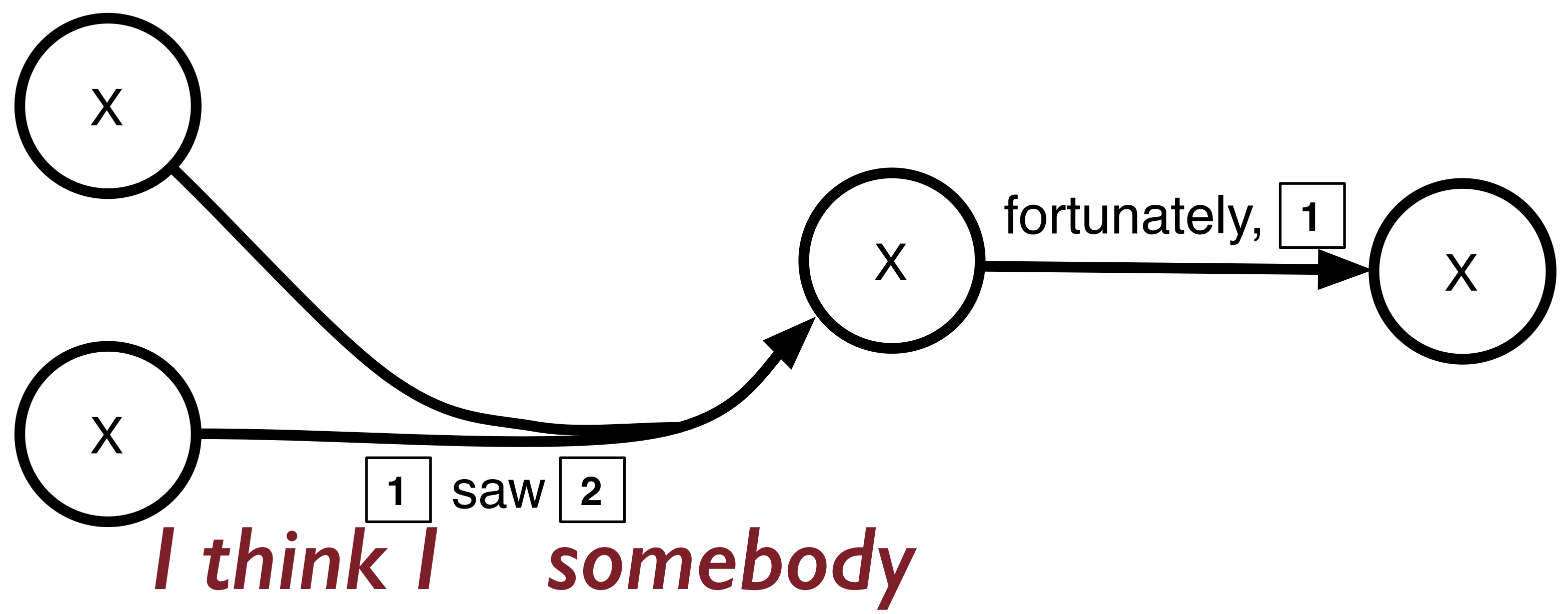


Two problems:

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?

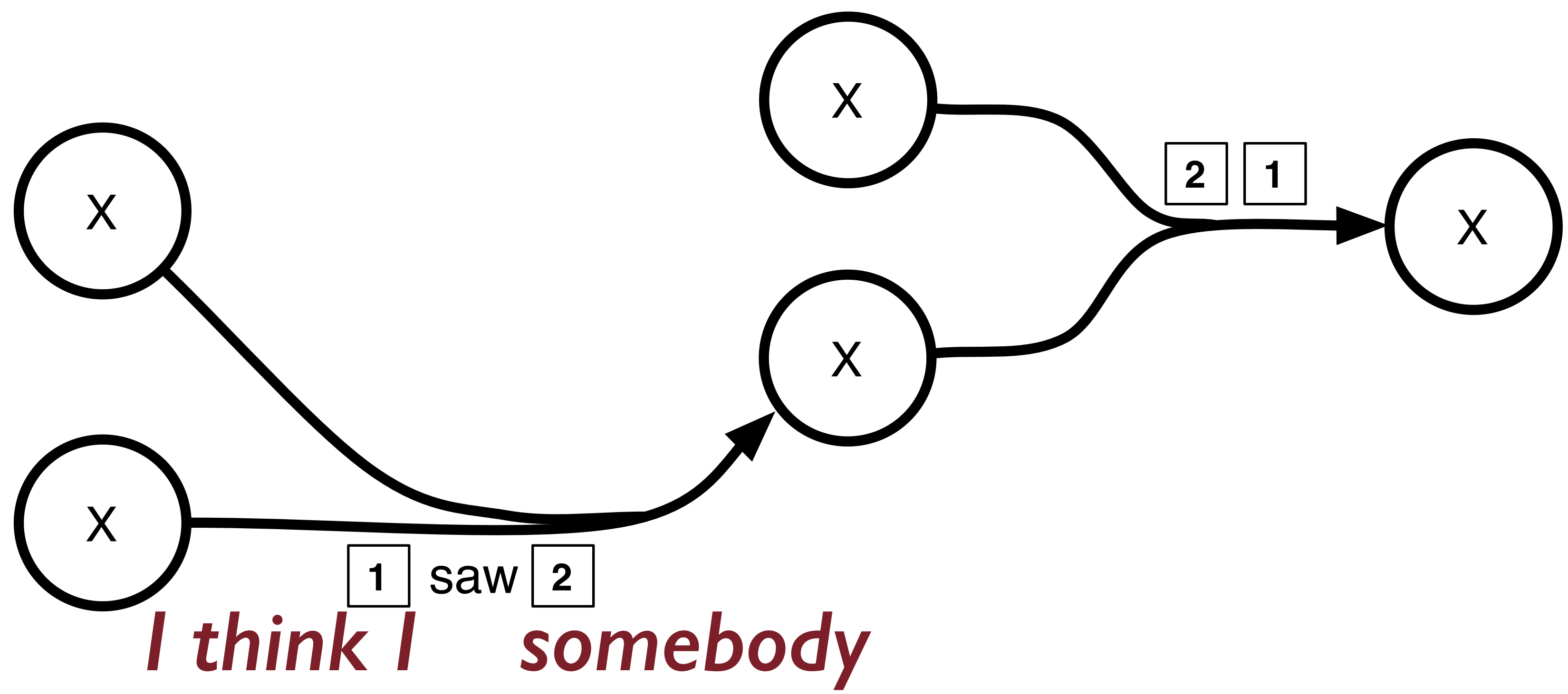


Two problems:

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.

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

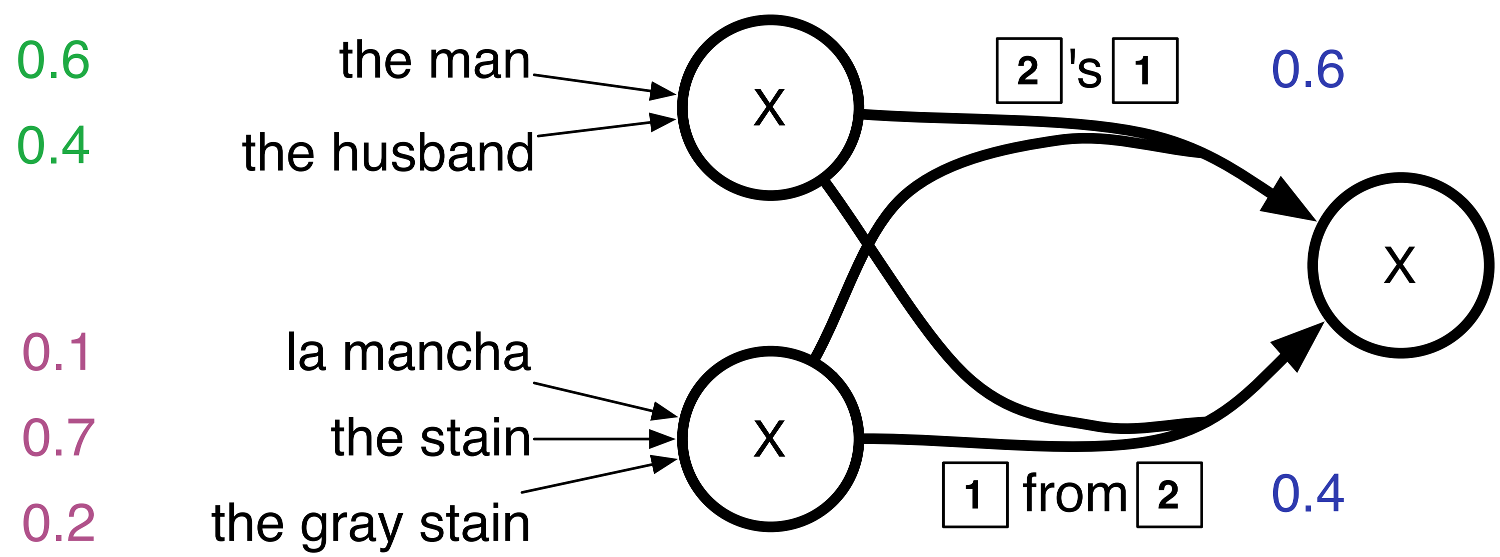
of

  - $(n\text{-}1)$ 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

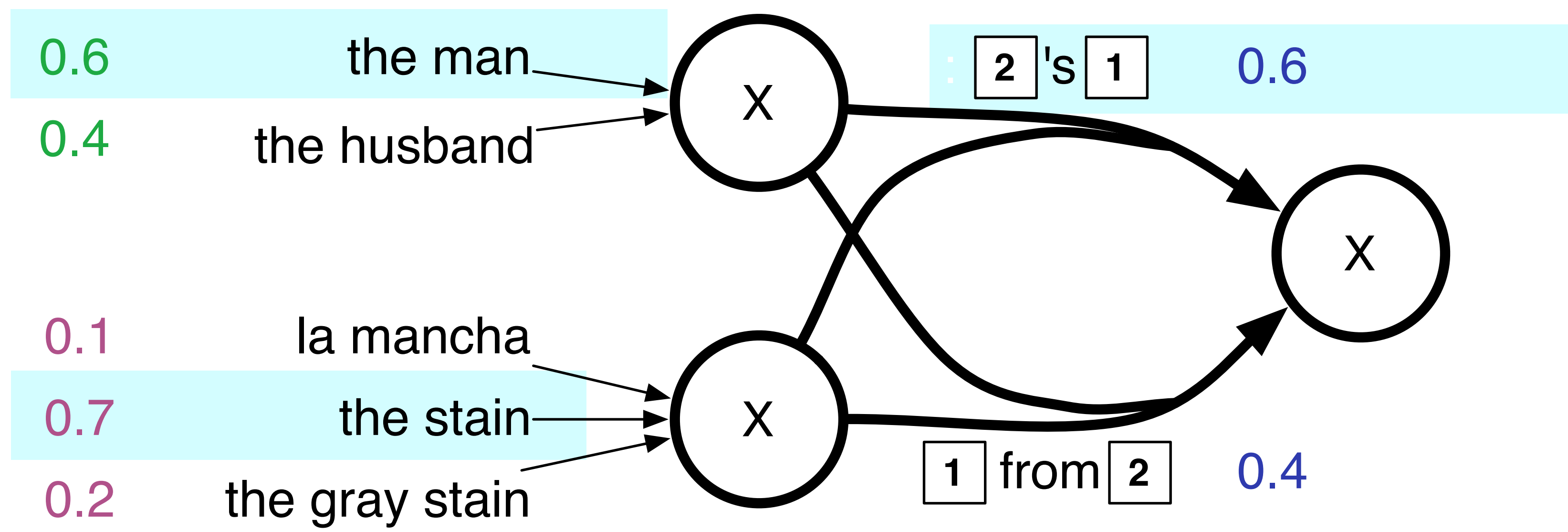
Hypergraph restructuring

- 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

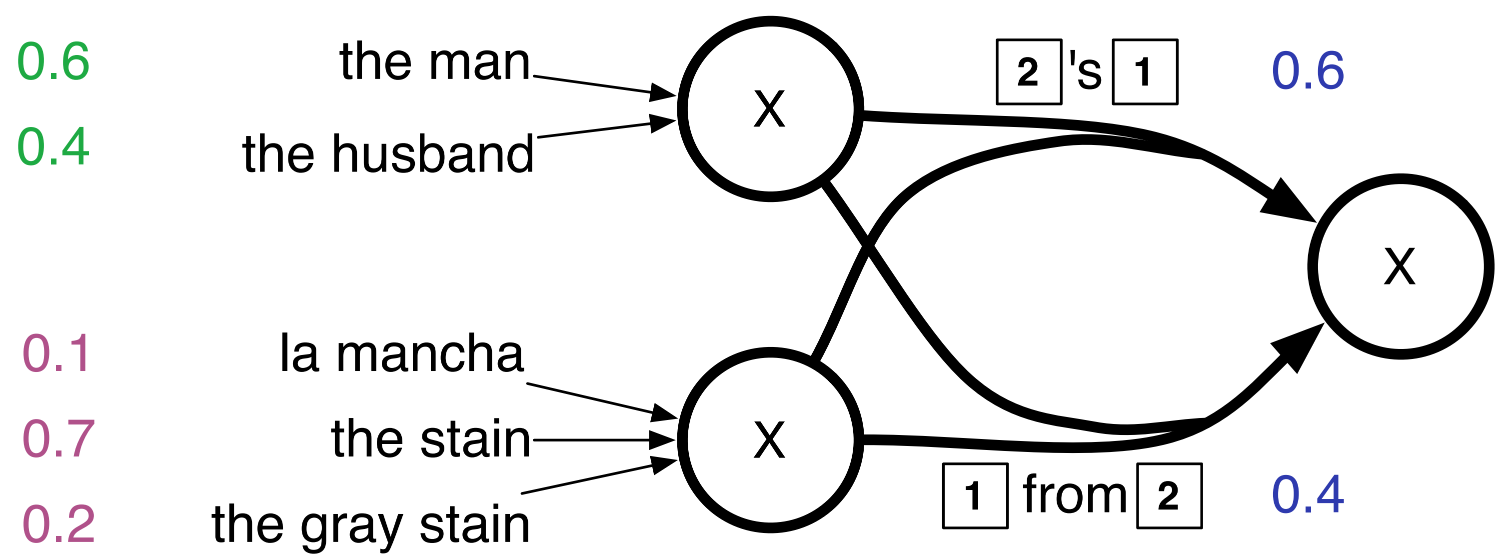


-LM Viterbi:

the stain's the man

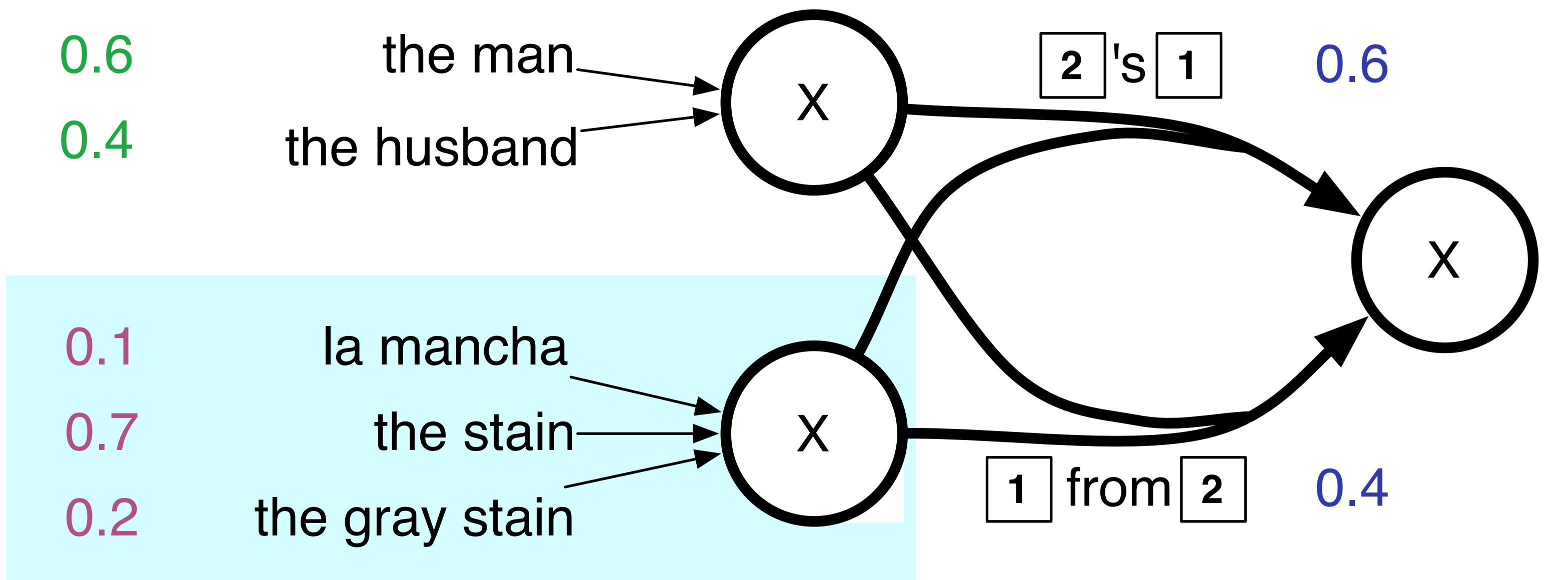
Hypergraph restructuring

Let's add a bi-gram language model!

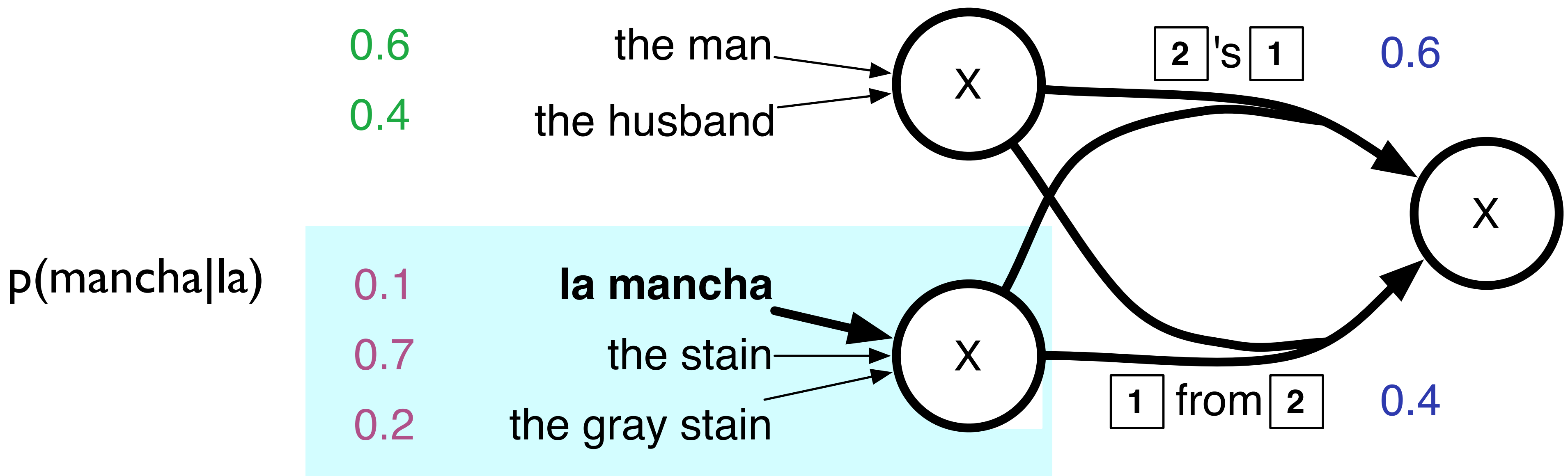


Hypergraph restructuring

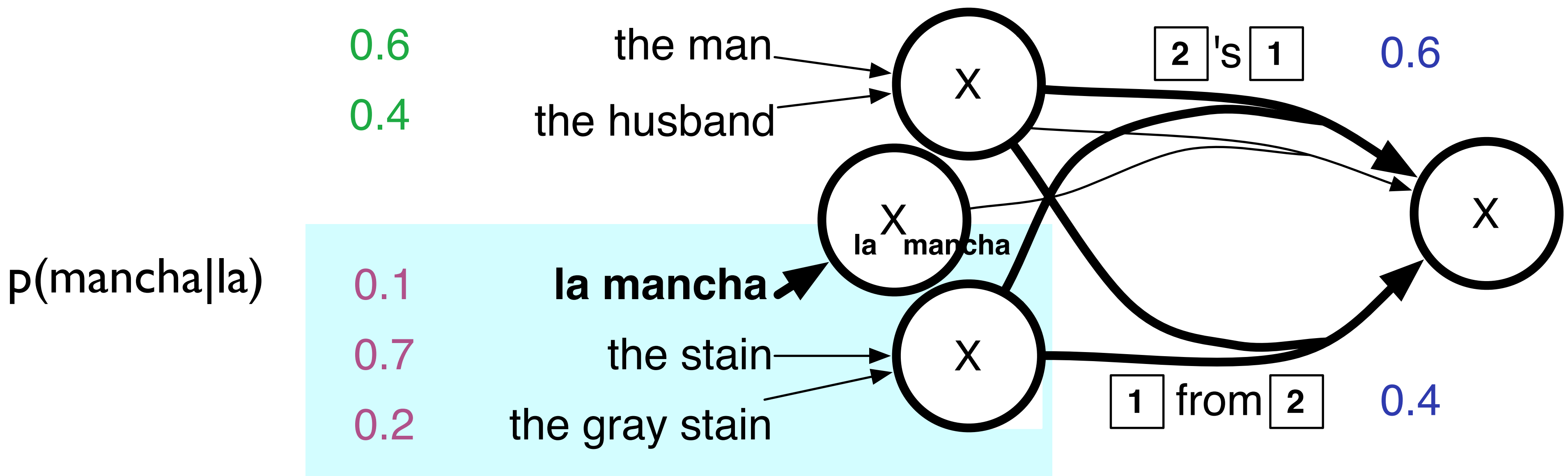
Let's add a bi-gram language model!



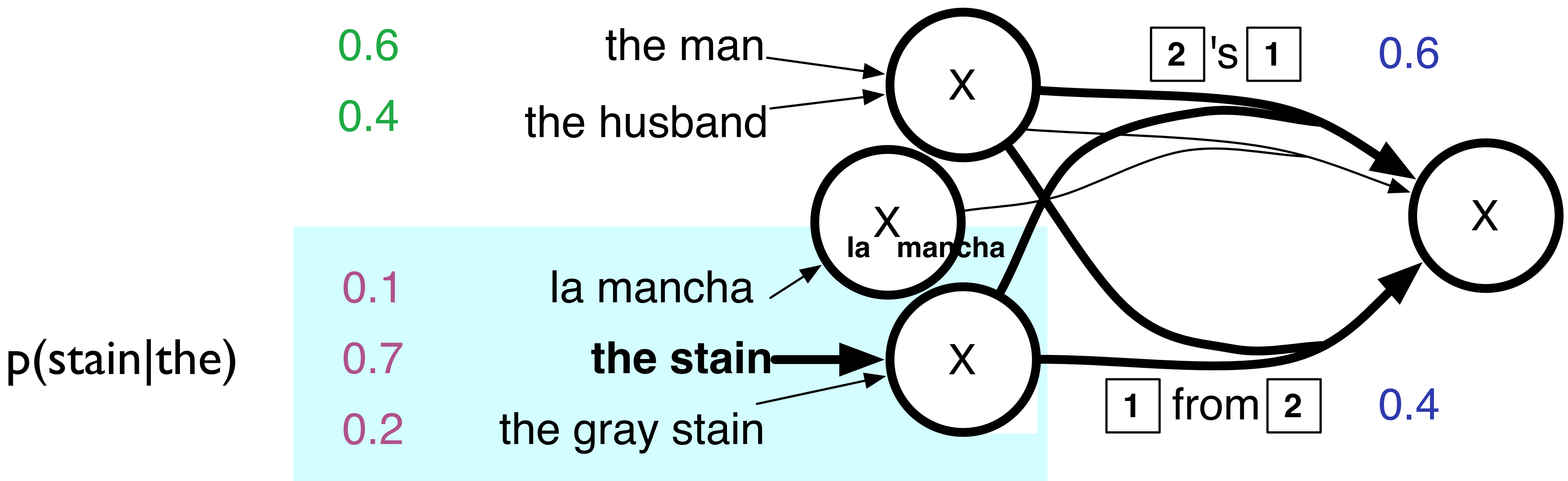
Hypergraph restructuring



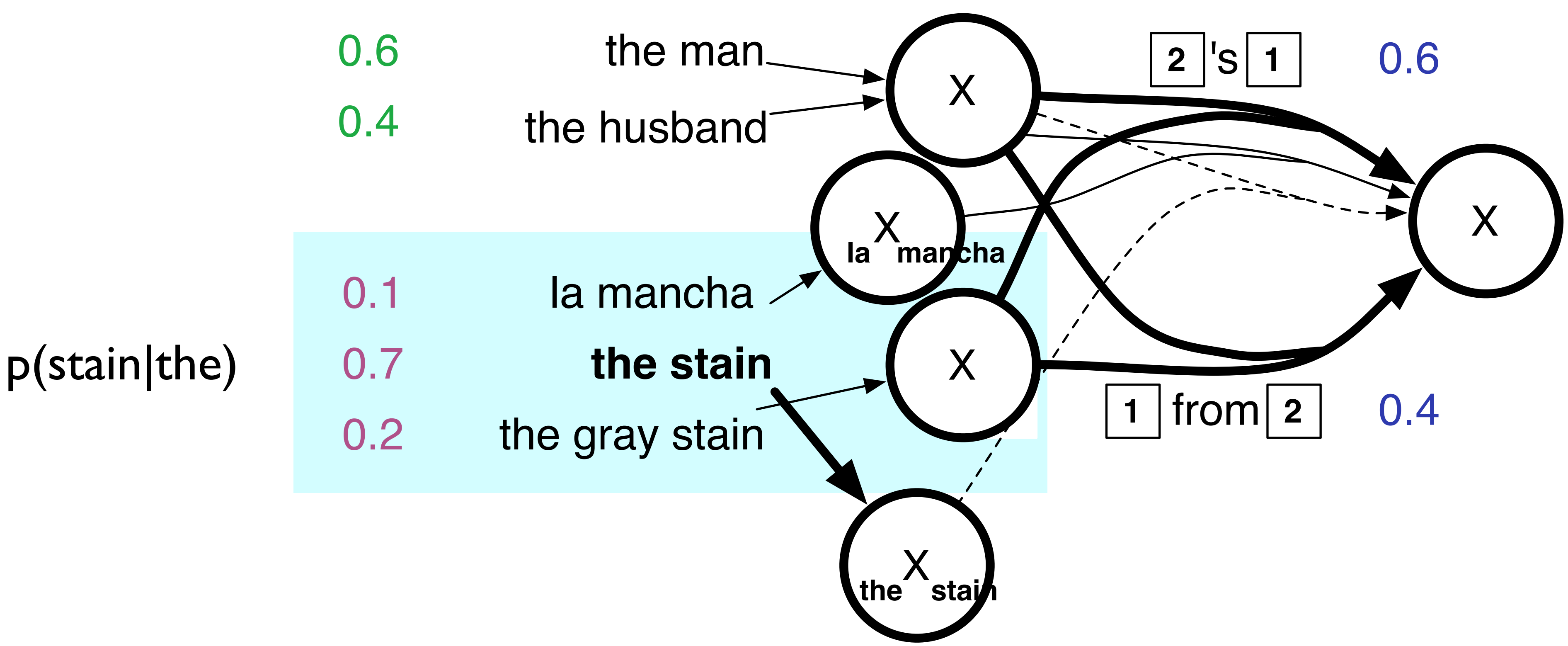
Hypergraph restructuring



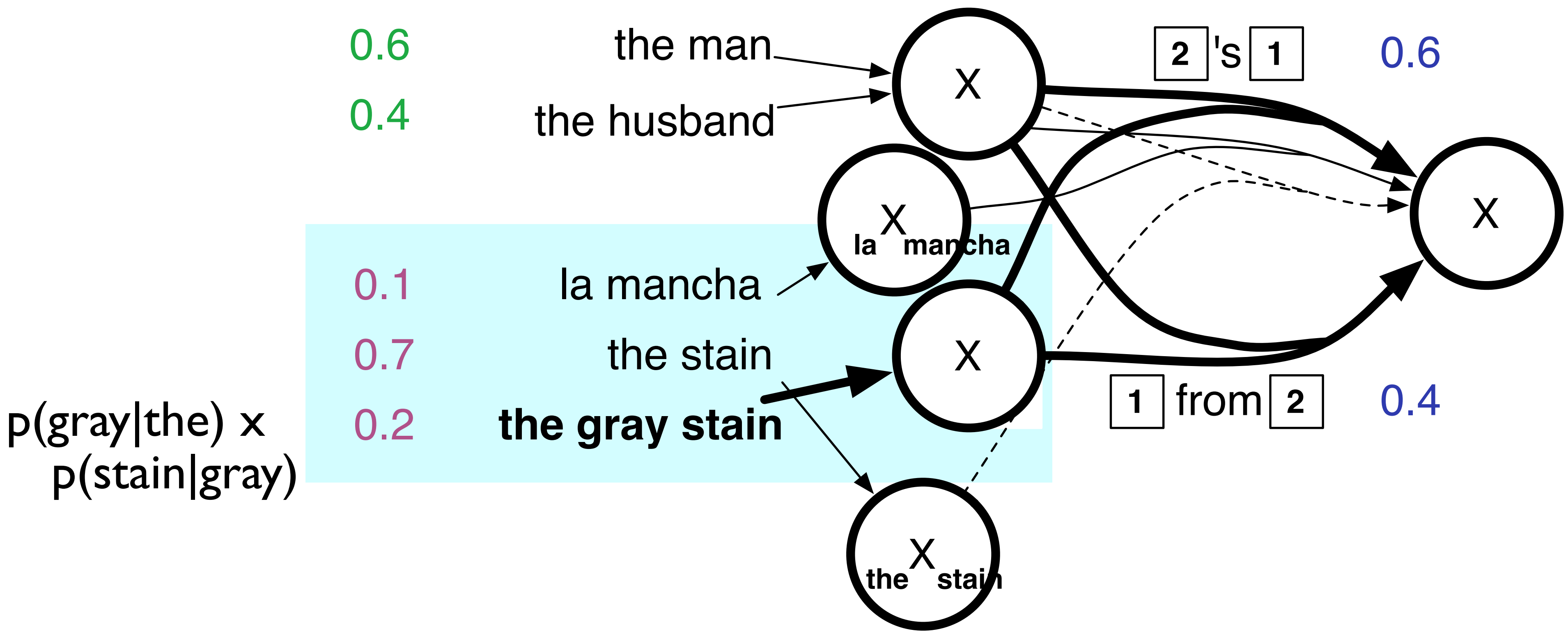
Hypergraph restructuring



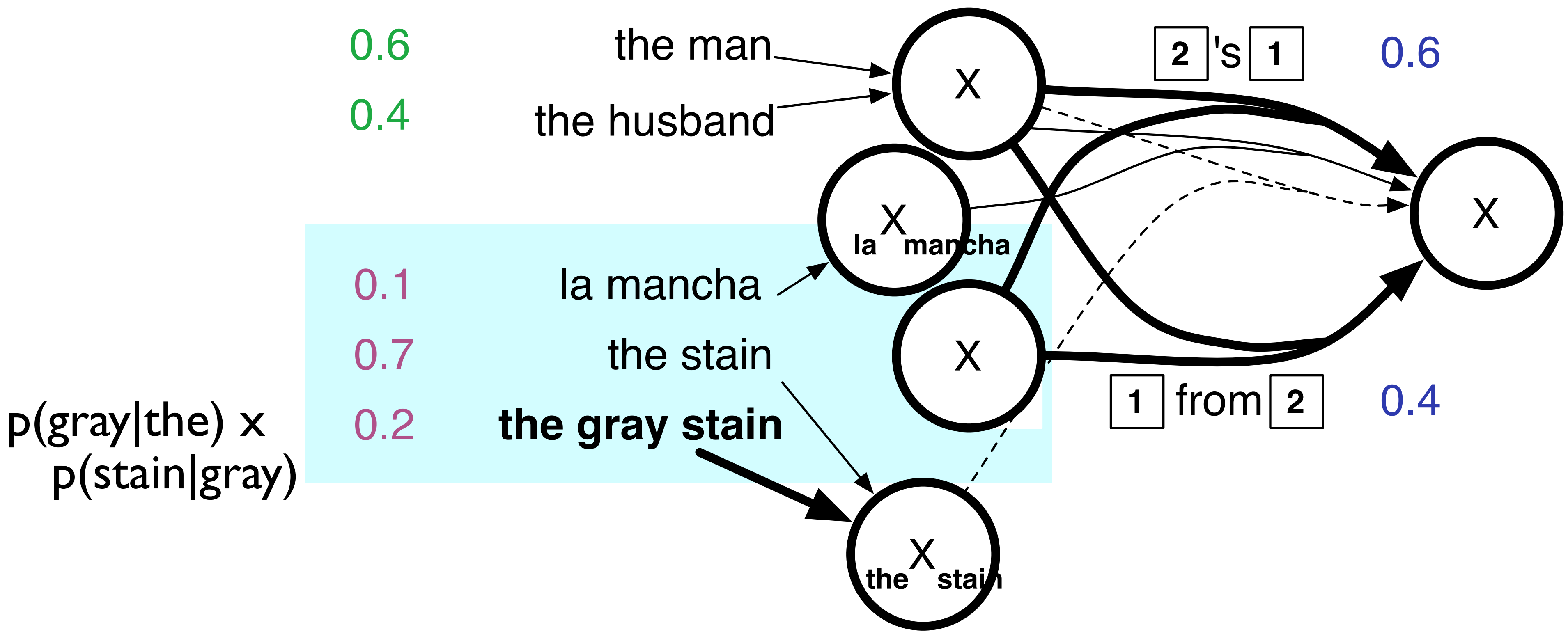
Hypergraph restructuring



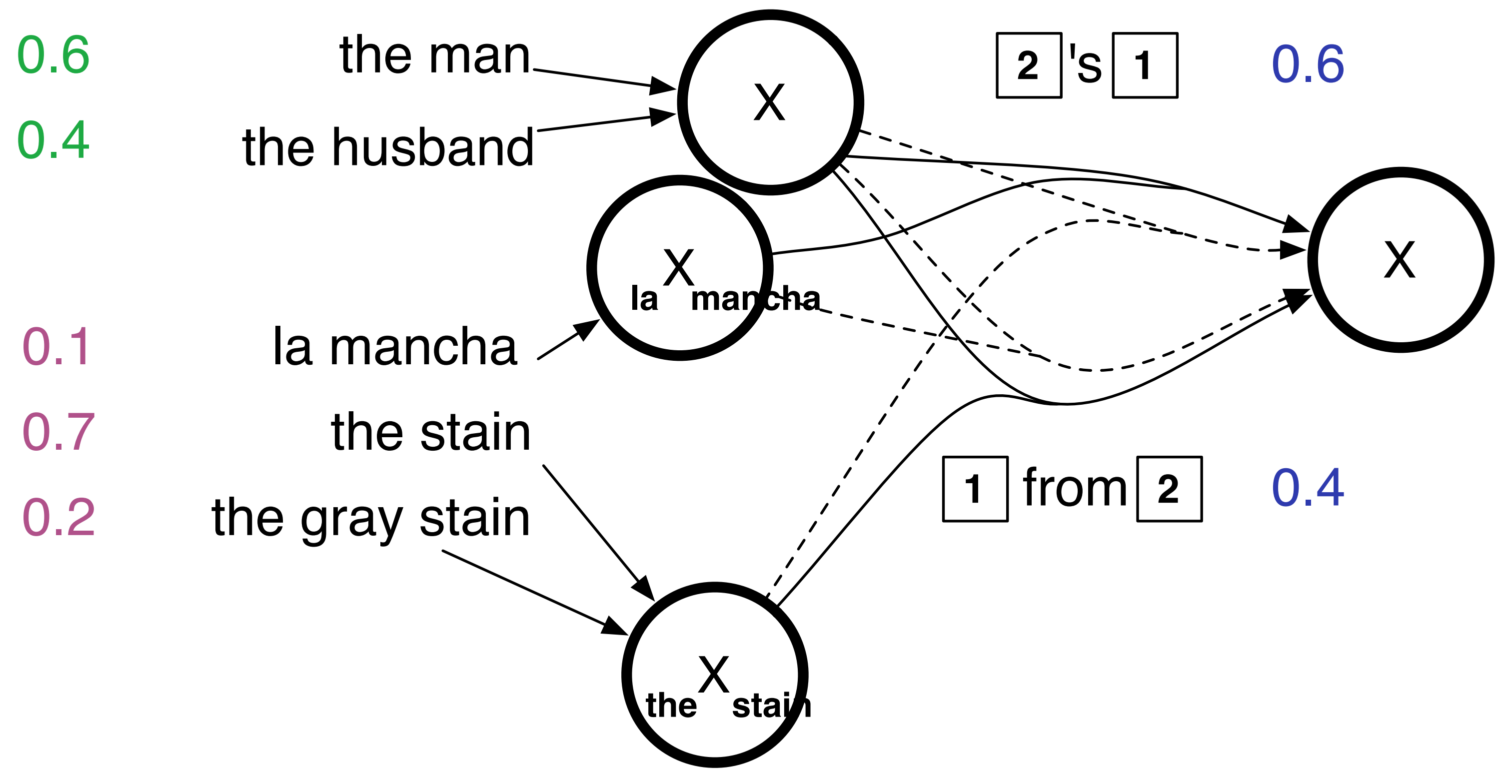
Hypergraph restructuring



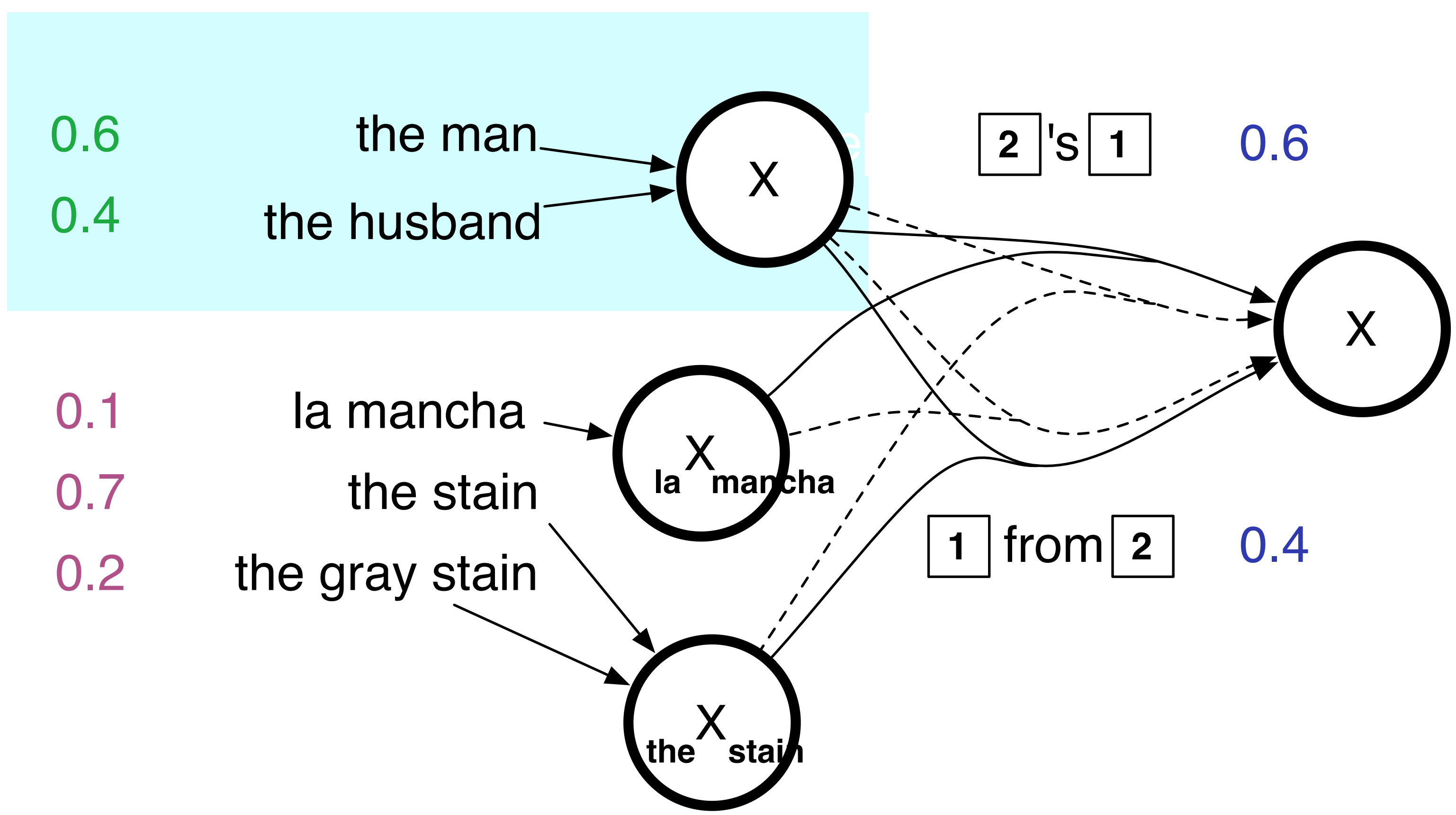
Hypergraph restructuring



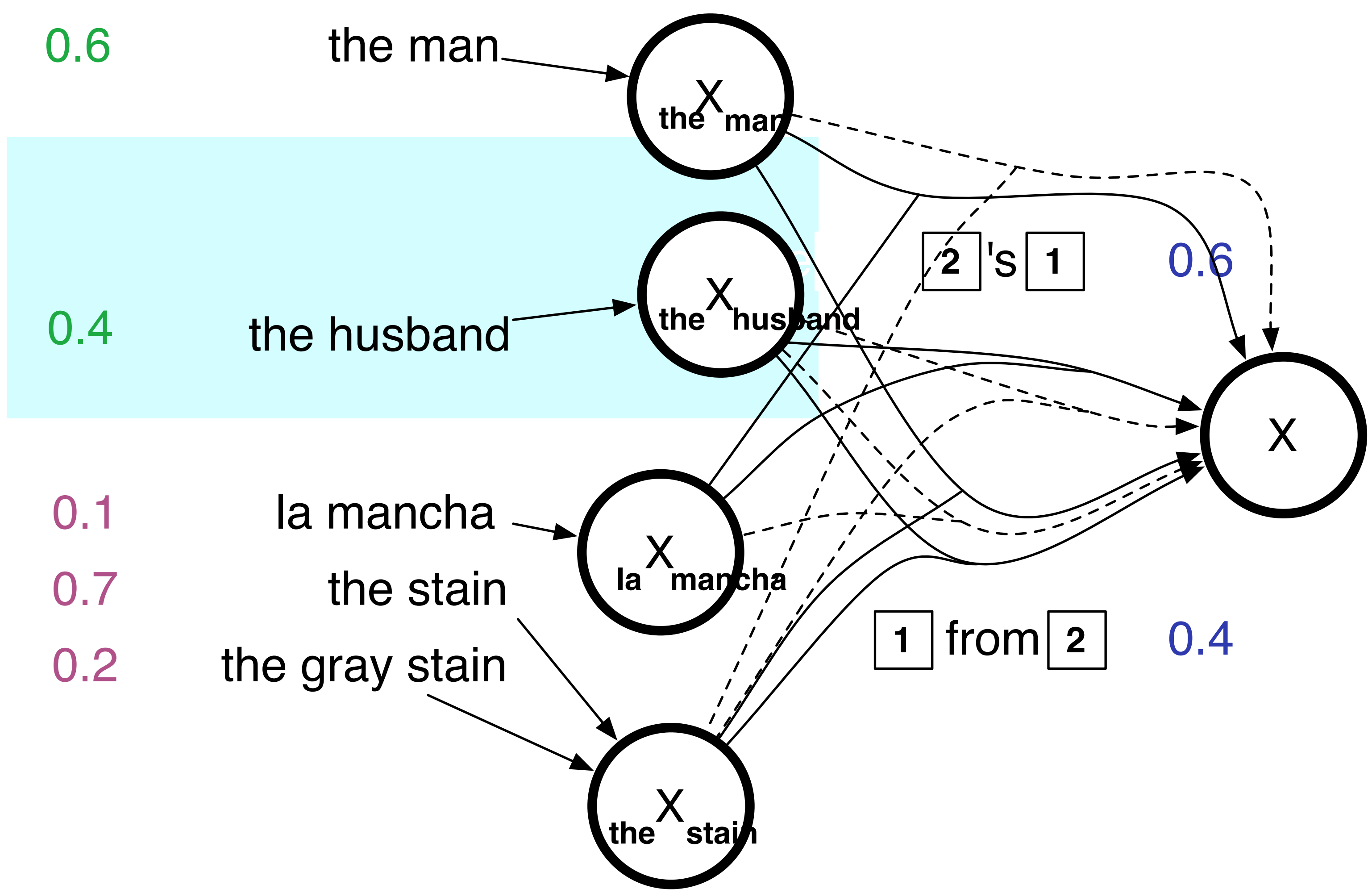
Hypergraph restructuring



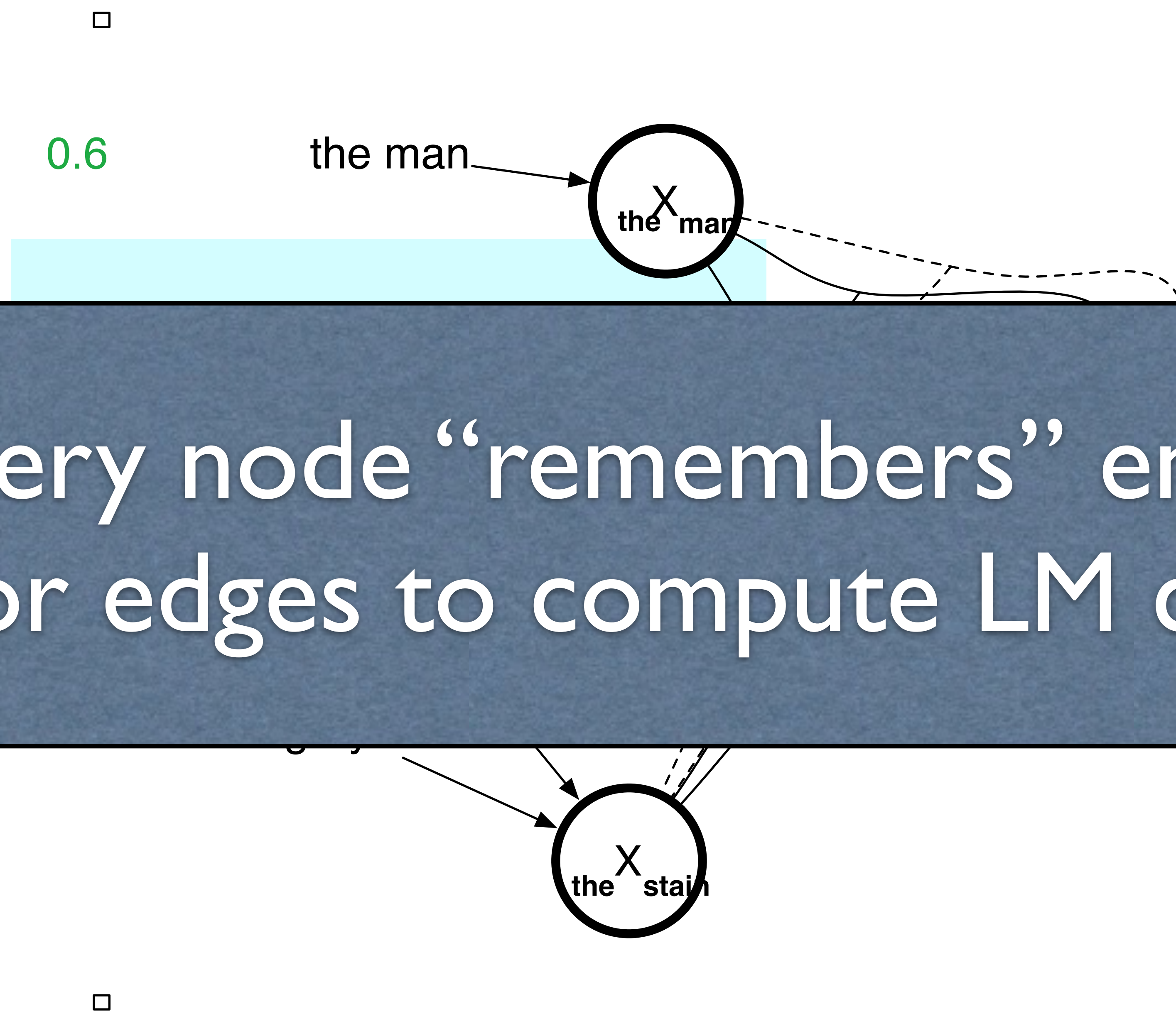
Hypergraph restructuring



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!

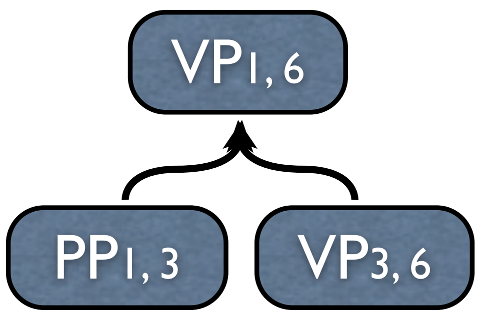
Cube pruning

- 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!

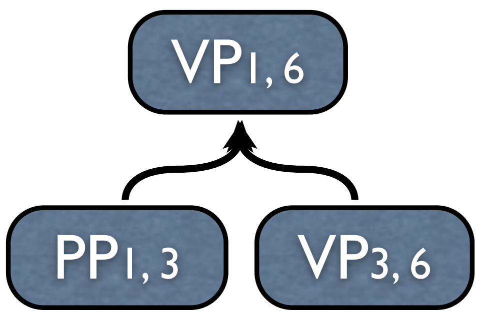
Cube Pruning



monotonic grid?

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

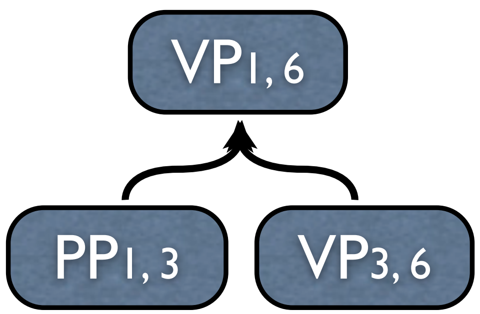
Cube Pruning



non-monotonic grid
due to LM combo costs

		$(\mathrm{PP}^{\,\text{with} \,\star\, \text{Sharon}}_{1,3})$	$(\mathrm{PP}^{\,\text{along} \,\star\, \text{Sharon}}_{1,3})$	$(\mathrm{PP}^{\,\text{with} \,\star\, \text{Shalong}}_{1,3})$
		1.0	3.0	8.0
$(\mathrm{VP}^{\,\text{held} \,\star\, \text{meeting}}_{3,6})$	1.0	2.0 + 0.5	4.0 + 5.0	9.0 + 0.5
$(\mathrm{VP}^{\,\text{held} \,\star\, \text{talk}}_{3,6})$	1.1	2.1 + 0.3	4.1 + 5.4	9.1 + 0.3
$(\mathrm{VP}^{\,\text{hold} \,\star\, \text{conference}}_{3,6})$	3.5	4.5 + 0.6	6.5 +10.5	11.5 + 0.6

Cube Pruning

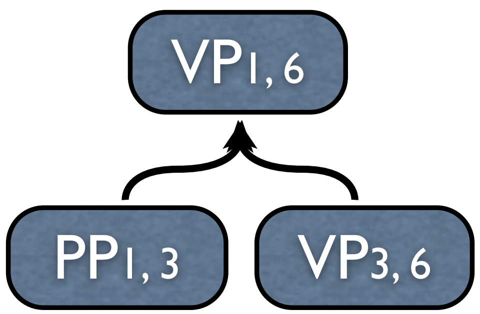


bigram (meeting, with)

non-monotonic grid
due to LM combo costs

		$(\mathrm{PP}_{1,3}^{\text{with} \star \text{Sharon}})$	$(\mathrm{PP}_{1,3}^{\text{along} \star \text{Sharon}})$	$(\mathrm{PP}_{1,3}^{\text{with} \star \text{Shalong}})$
		1.0	3.0	8.0
$(\mathrm{VP}_{3,6}^{\text{held} \star \text{meeting}})$	1.0	2.0 + 0.5	4.0 + 5.0	9.0 + 0.5
$(\mathrm{VP}_{3,6}^{\text{held} \star \text{talk}})$	1.1	2.1 + 0.3	4.1 + 5.4	9.1 + 0.3
$(\mathrm{VP}_{3,6}^{\text{hold} \star \text{conference}})$	3.5	4.5 + 0.6	6.5 + 10.5	11.5 + 0.6

Cube Pruning



non-monotonic grid
due to LM combo costs

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

		$(\mathrm{PP}^{\,\text{with} \,\star\, \text{Sharon}}_{1,3})$	$(\mathrm{PP}^{\,\text{along} \,\star\, \text{Sharon}}_{1,3})$	$(\mathrm{PP}^{\,\text{with} \,\star\, \text{Shalong}}_{1,3})$
		1.0	3.0	8.0
$(\mathrm{VP}^{\,\text{held} \,\star\, \text{meeting}}_{3,6})$	1.0	2.5	9.0	9.5
$(\mathrm{VP}^{\,\text{held} \,\star\, \text{talk}}_{3,6})$	1.1	2.4	9.5	9.4
$(\mathrm{VP}^{\,\text{hold} \,\star\, \text{conference}}_{3,6})$	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
- push the two successors

		$(\mathrm{PP}^{\text{with} \star \text{Sharon}}_{1,3})$	$(\mathrm{PP}^{\text{along} \star \text{Sharon}}_{1,3})$	$(\mathrm{PP}^{\text{with} \star \text{Shalong}}_{1,3})$
		1.0	3.0	8.0
$(\mathrm{VP}^{\text{held} \star \text{meeting}}_{3,6})$	1.0	2.5	9.0	9.5
$(\mathrm{VP}^{\text{held} \star \text{talk}}_{3,6})$	1.1	2.4	9.5	9.4
$(\mathrm{VP}^{\text{hold} \star \text{conference}}_{3,6})$	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
- push the two successors

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

LM Integration

Method	Settings	Time	BLEU
rescore	$k = 10^4$	16	33.31
rescore	$k = 10^5$	139	33.33
intersect*		1455	37.09
cube prune	$\varepsilon = 0$	23	36.14
cube prune	$\varepsilon = 0.1$	35	36.77
cube prune	$\varepsilon = 0.2$	111	36.91