Phrase-Based MT: Decoding

February 7, 2013

Phrase Based MT

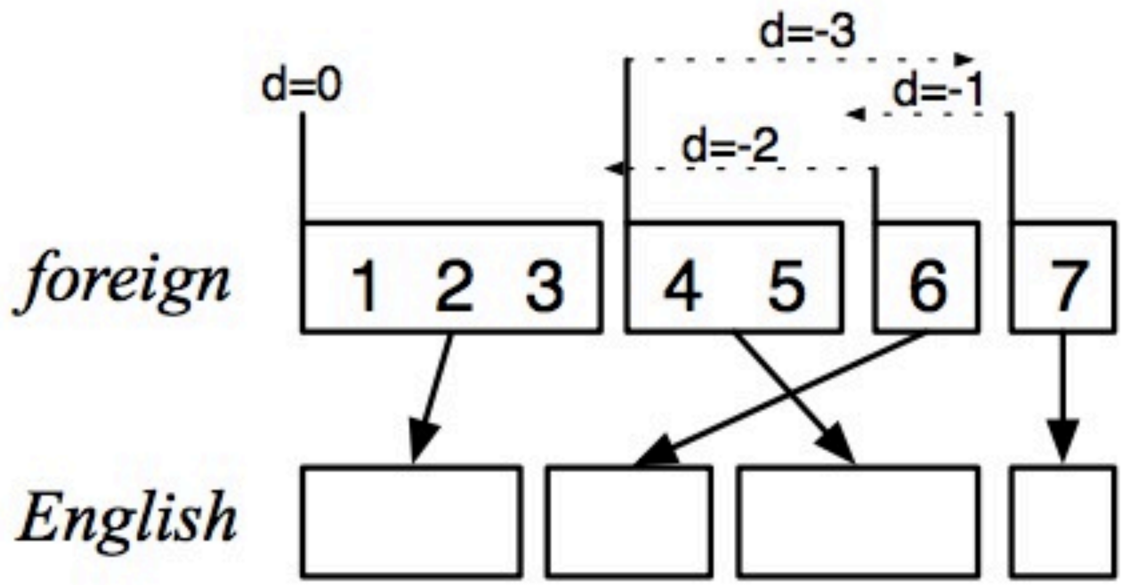
$$\begin{aligned}\mathbf{e}^* &= \arg\max_{\mathbf{e}} p(\mathbf{e} \mid \mathbf{f}) \\ &= \arg\max_{\mathbf{e}} p(\mathbf{f} \mid \mathbf{e}) \times p(\mathbf{e}) \\ &\approx \arg\max_{\mathbf{e}} p(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \times p(\mathbf{e})\end{aligned}$$

- Recipe
  - Segmentation / Alignment model
  - Phrase model
  - Language Model

Phrase Tables

$\bar{\mathbf{f}}$	$\bar{\mathbf{e}}$	$p(\bar{\mathbf{f}} \mid \bar{\mathbf{e}})$
das Thema	the issue	0.41
	the point	0.72
	the subject	0.47
	the thema	0.99
es gibt	there is	0.96
	there are	0.72
morgen	tomorrow	0.9
fliege ich	will I fly	0.63
	will fly	0.17
	I will fly	0.13

Reordering Model



phrase	translates	movement	distance
1	1–3	start at beginning	0
2	6	skip over 4–5	+2
3	4–5	move back over 4–6	-3
4	7	skip over 6	+1

Scoring function: $d(x) = \alpha^{|x|}$ — exponential with distance

Translation Process

- Task: translate this sentence from German into English

er geht ja nicht nach hause

Translation Process

- Task: translate this sentence from German into English

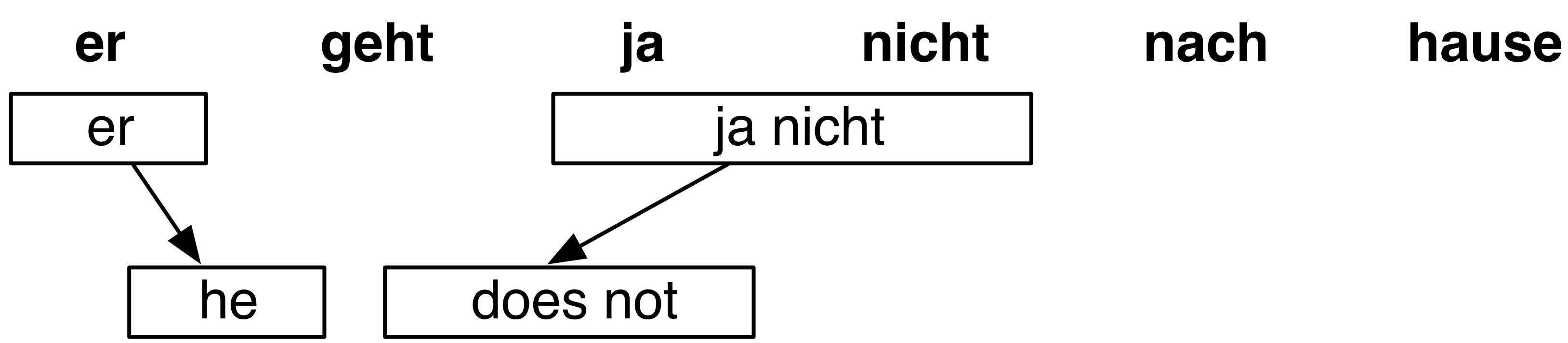
er **geht** **ja** **nicht** **nach** **hause**

er → he

- Pick phrase in input, translate

Translation Process

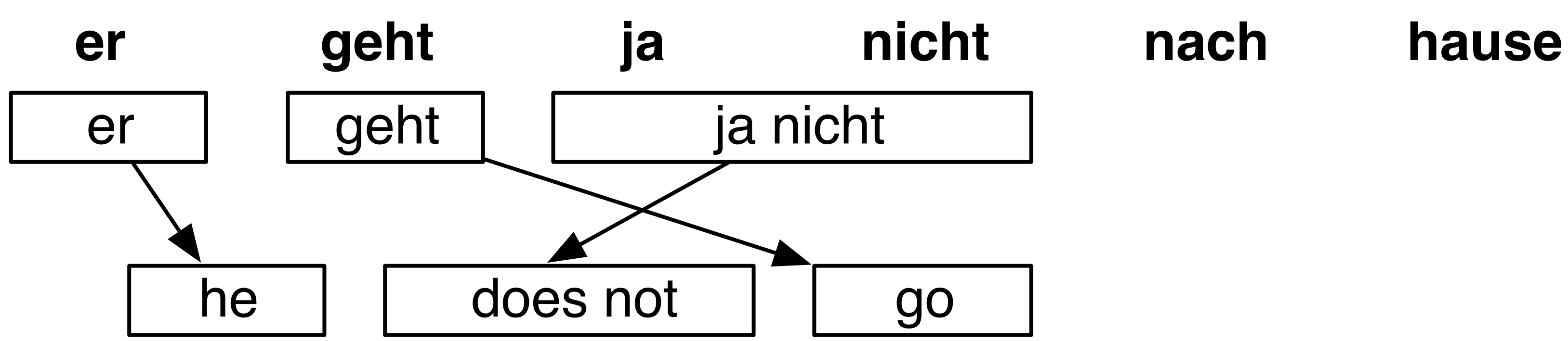
- Task: translate this sentence from German into English



- Pick phrase in input, translate
  - it is allowed to pick words out of sequence reordering
  - phrases may have multiple words: many-to-many translation

Translation Process

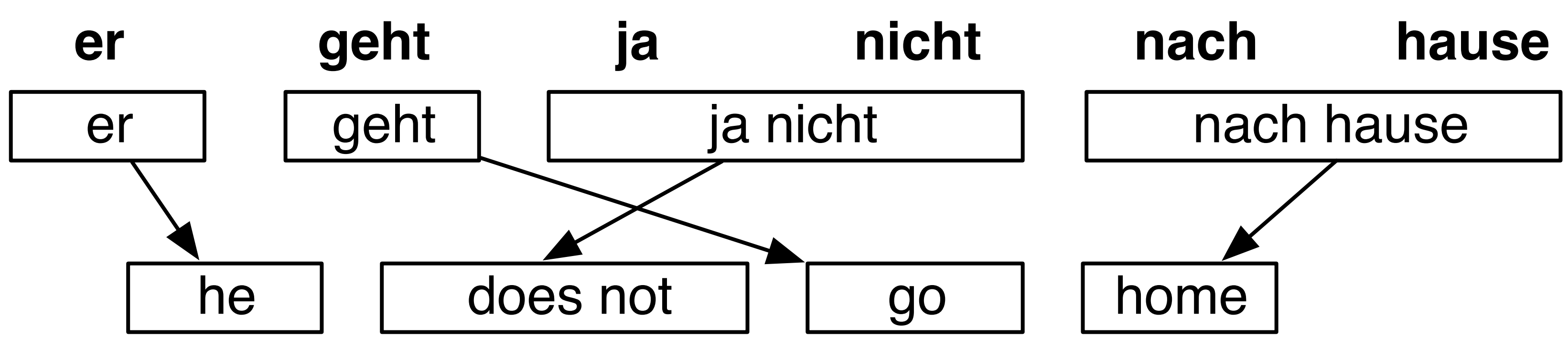
- Task: translate this sentence from German into English



- Pick phrase in input, translate

Translation Process

- Task: translate this sentence from German into English



- Pick phrase in input, translate

Computing Translation Probability

- Probabilistic model for phrase-based translation:

$$\mathbf{e}_{\text{best}} = \text{argmax}_{\mathbf{e}} \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\; d(start_i - end_{i-1} - 1)\; p_{\text{LM}}(\mathbf{e})$$

- Score is computed incrementally for each partial hypothesis

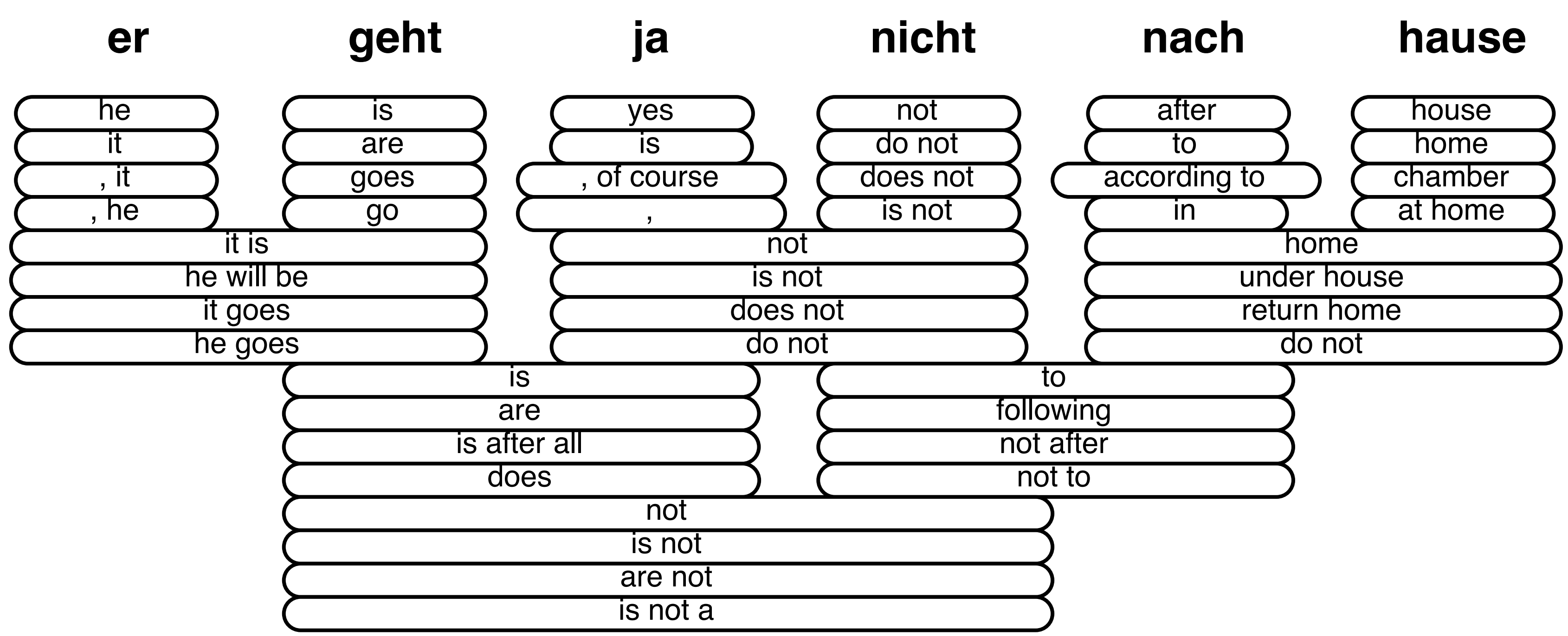
- Components

Phrase translation Picking phrase $\bar{f}_i$ to be translated as a phrase $\bar{e}_i$
→ look up score $\phi(\bar{f}_i|\bar{e}_i)$ from phrase translation table

Reordering Previous phrase ended in end_{i-1}, current phrase starts at $start_i$
→ compute $d(start_i - end_{i-1} - 1)$

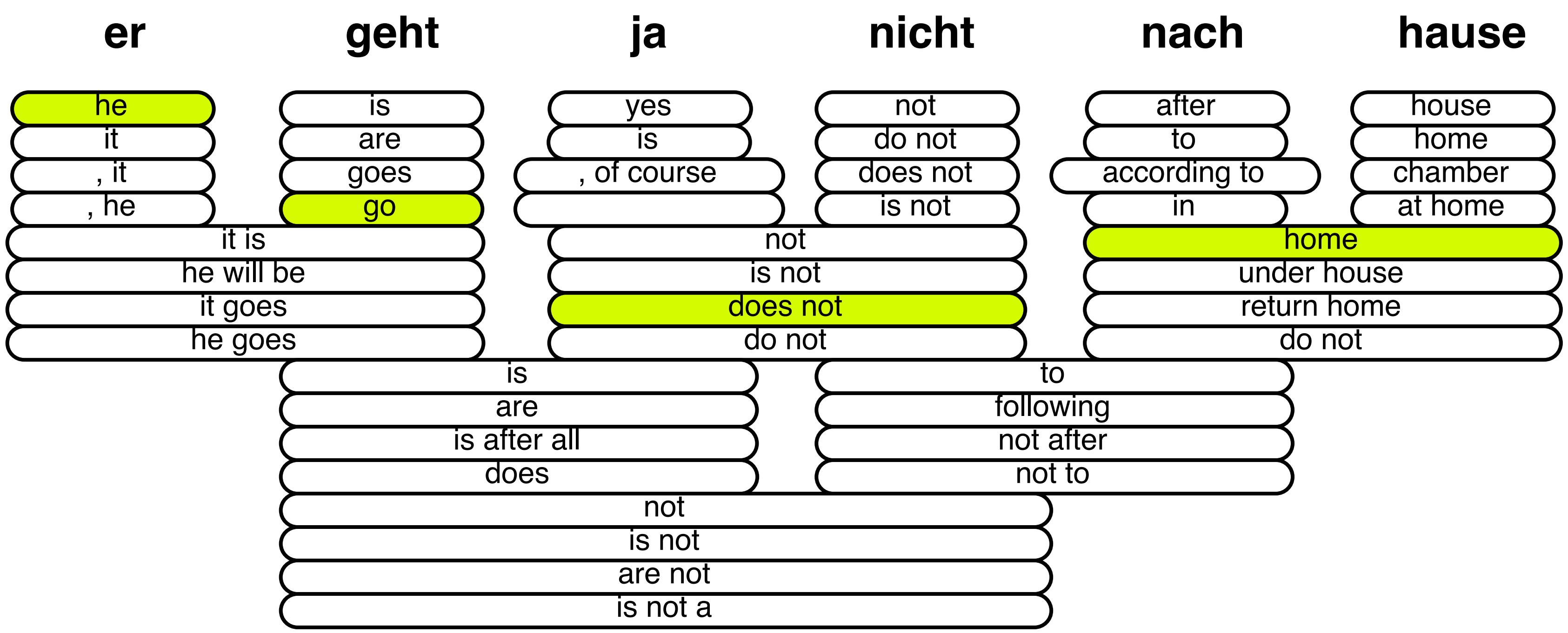
Language model For n-gram model, need to keep track of last $n-1$ words
→ compute score $p_{\text{LM}}(w_i|w_{i-(n-1)}, ..., w_{i-1})$ for added words w_i

Translation Options



- Many translation options to choose from
  - in Europarl phrase table: 2727 matching phrase pairs for this sentence
  - by pruning to the top 20 per phrase, 202 translation options remain

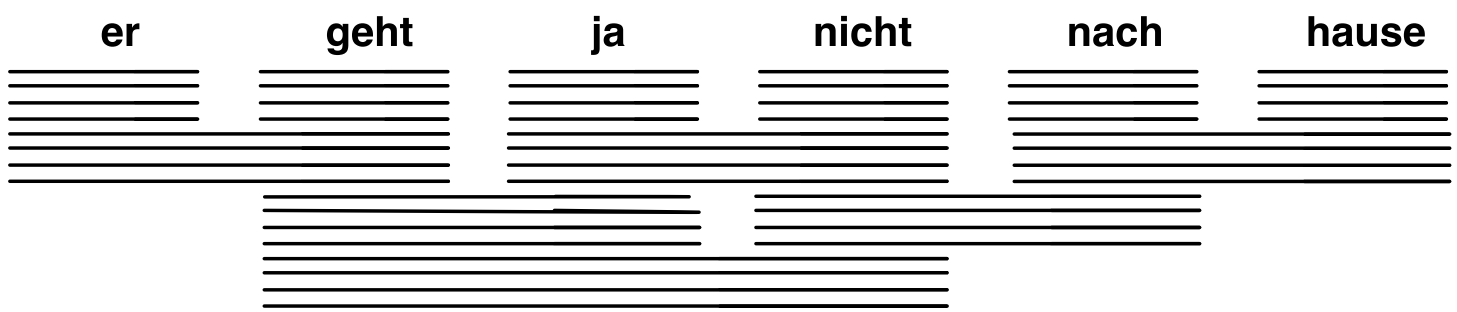
Translation Options



- The machine translation decoder does not know the right answer
  - picking the right translation options
  - arranging them in the right order

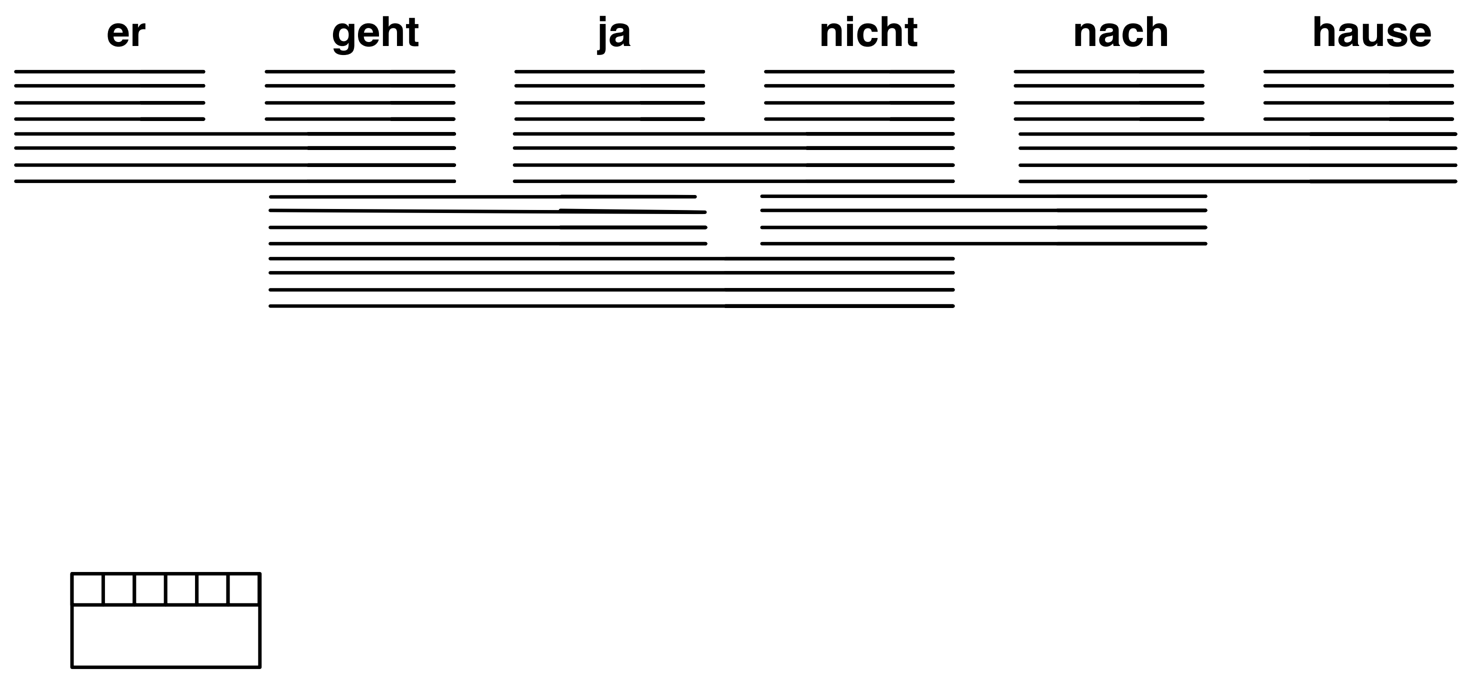
$\longrightarrow$ Search problem solved by heuristic beam search

Decoding: Precompute Translation Options



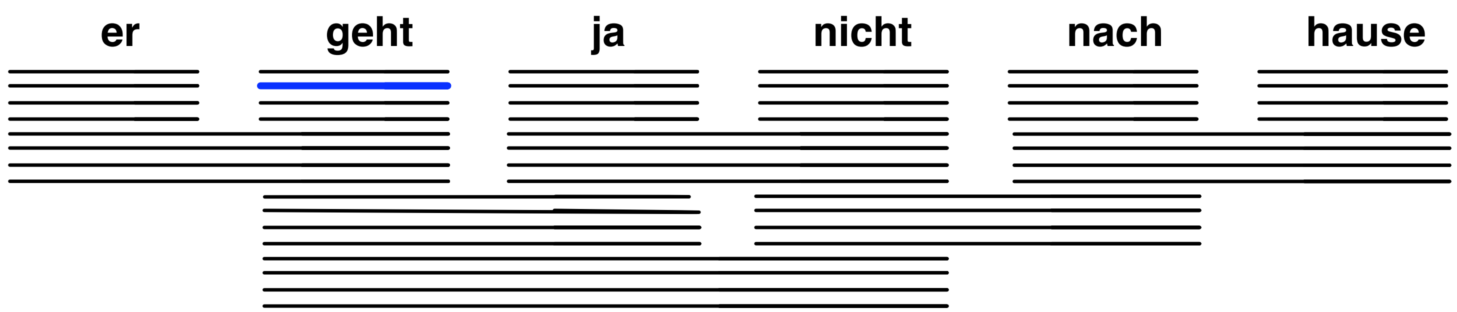
consult phrase translation table for all input phrases

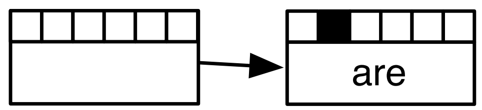
Decoding: Start with Initial Hypothesis



initial hypothesis: no input words covered, no output produced

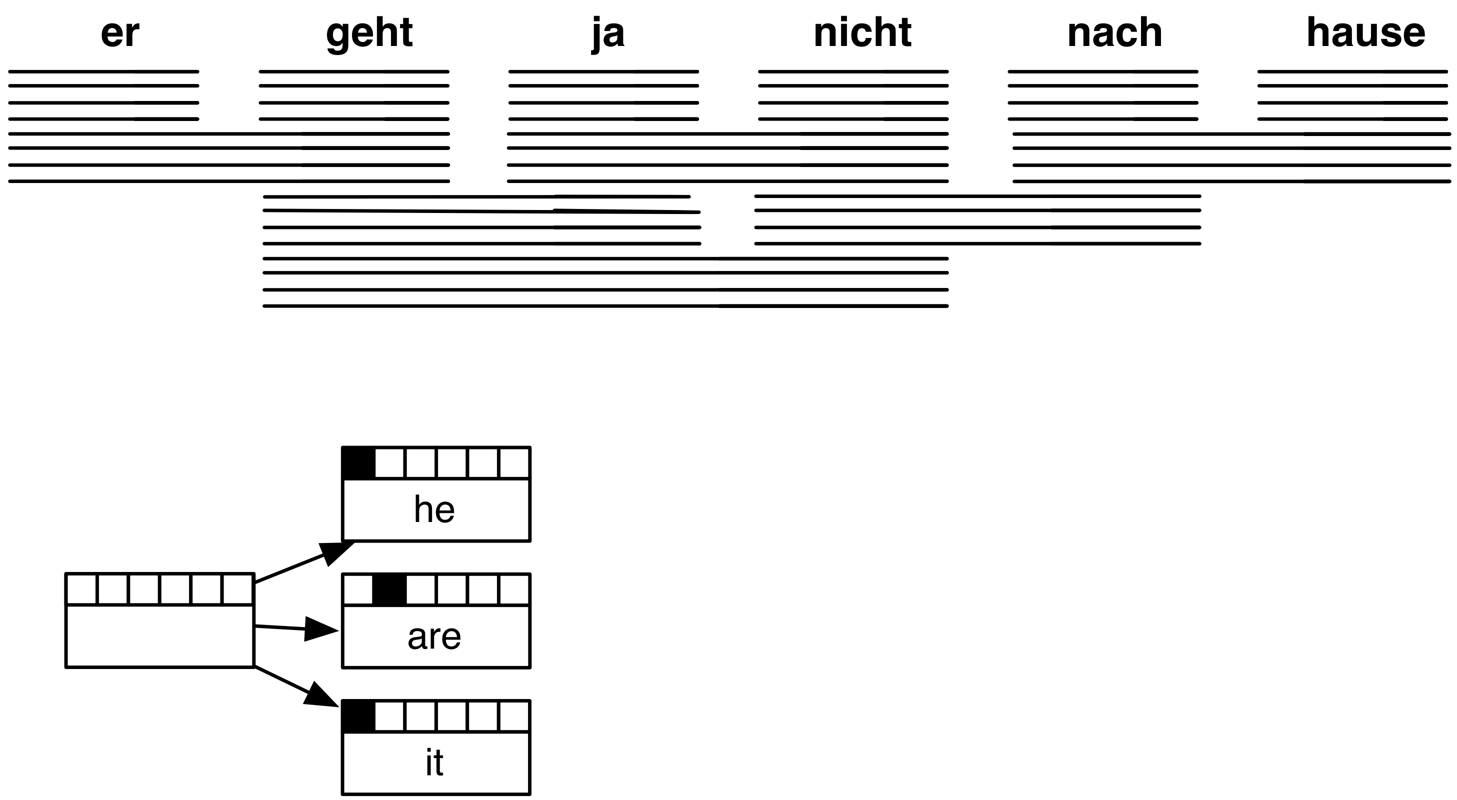
Decoding: Hypothesis Expansion





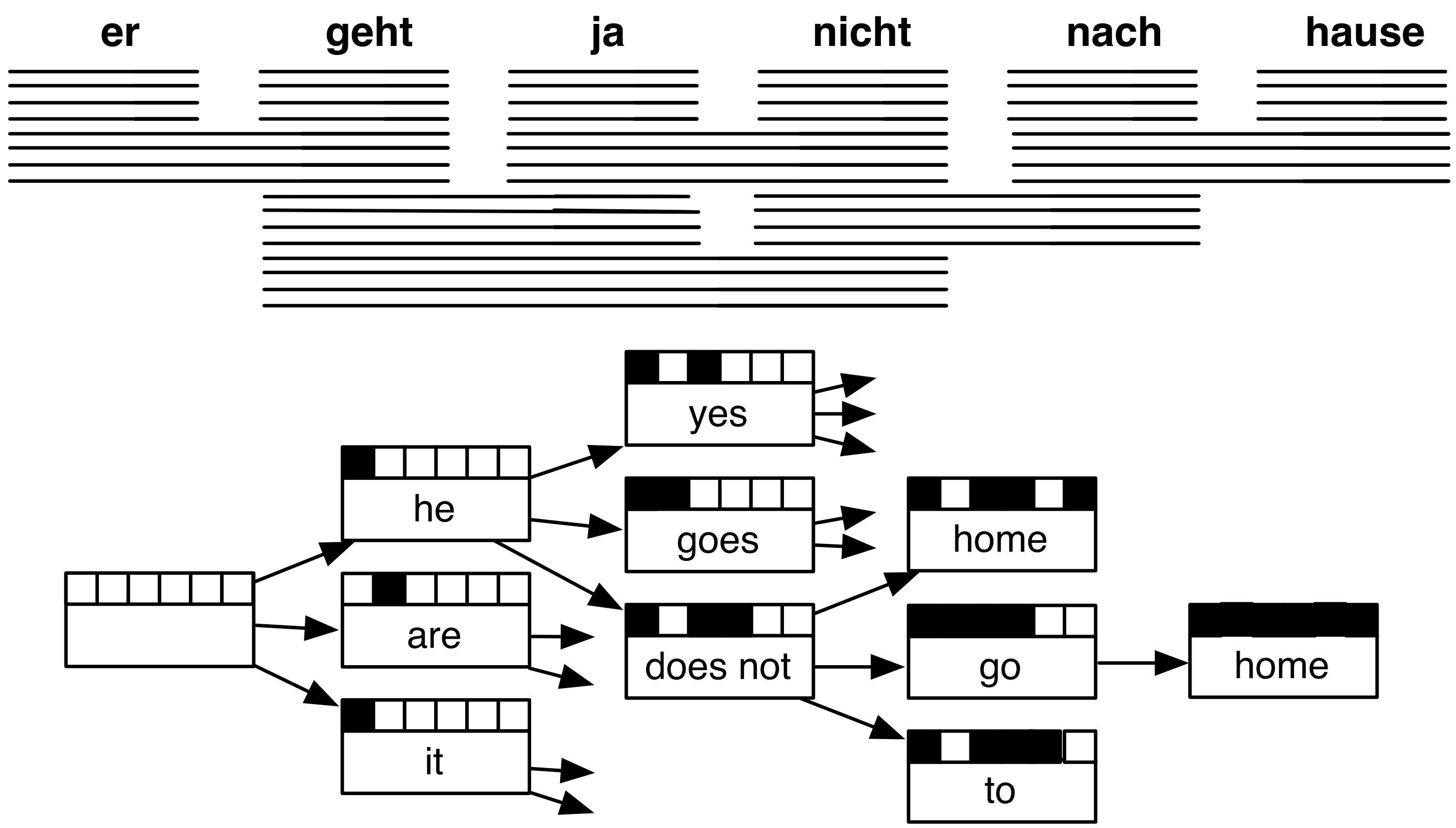
pick any translation option, create new hypothesis

Decoding: Hypothesis Expansion



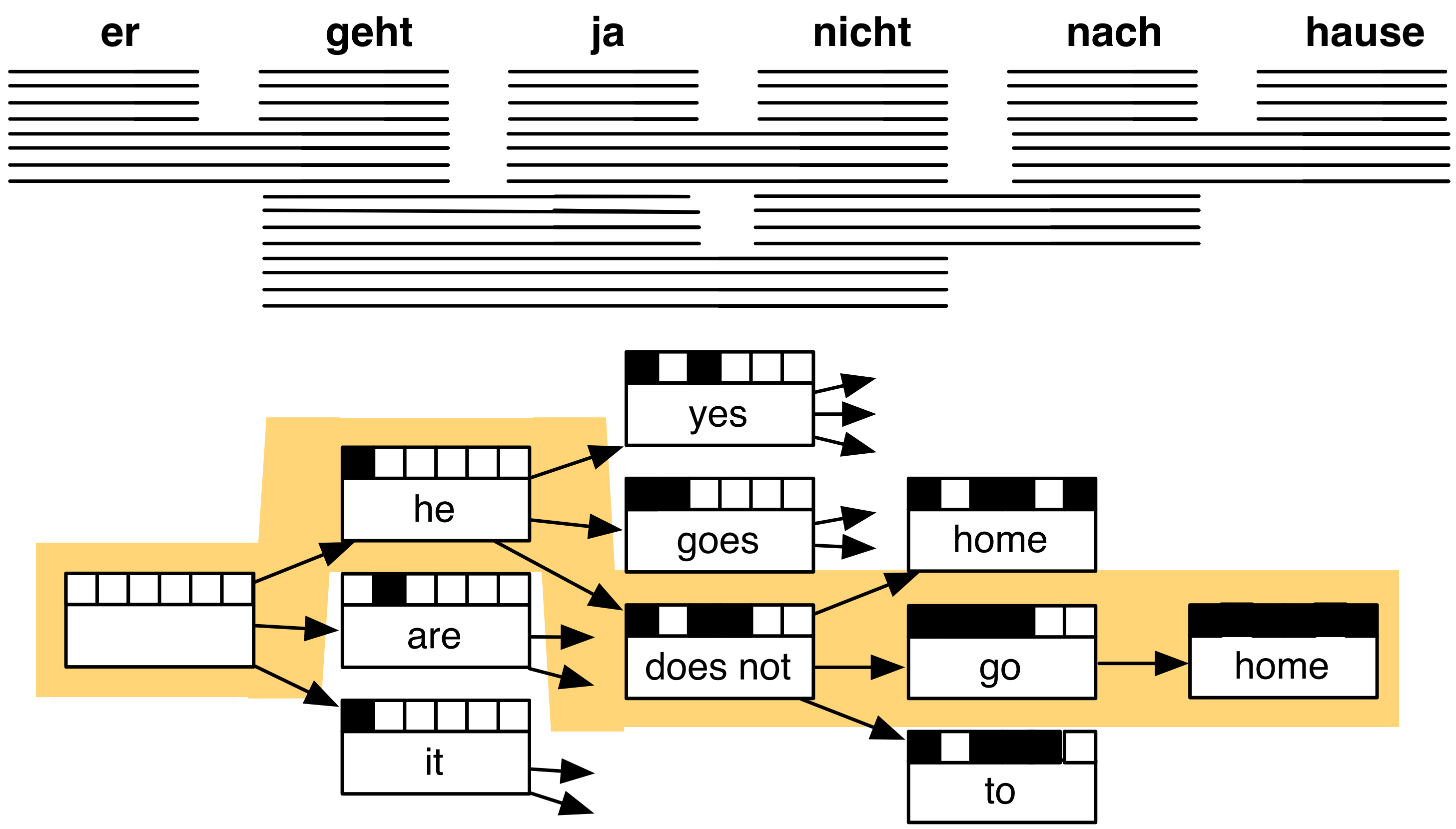
create hypotheses for all other translation options

Decoding: Hypothesis Expansion



also create hypotheses from created partial hypothesis

Decoding: Find Best Path



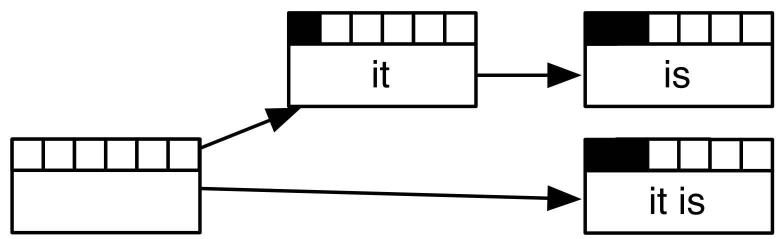
backtrack from highest scoring complete hypothesis

Complexity

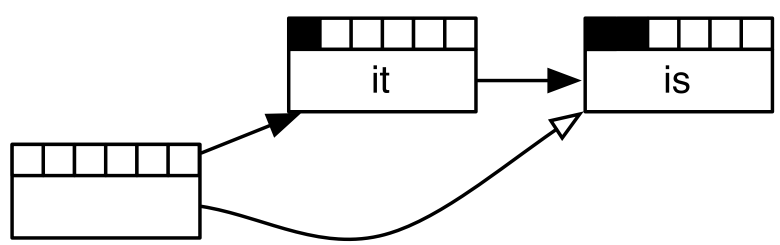
- This is an NP-complete problem
    - Reduction to TSP (sketch)
        - Each source word is a city
        - A bigram LM encodes the distance between pairs of cities
    - Knight (1999) has careful proof
- How do we solve such problems?
    - Dynamic programming [risk free]
        - The state is the current city C & the set of previous visited cities
        - Doesn't matter the order the previous list was visited in as long as we keep the best path to C through
        - How many states are there?
    - Approximate search [risky]

Recombination

- Two hypothesis paths lead to two matching hypotheses
  - same number of foreign words translated
  - same English words in the output
  - different scores

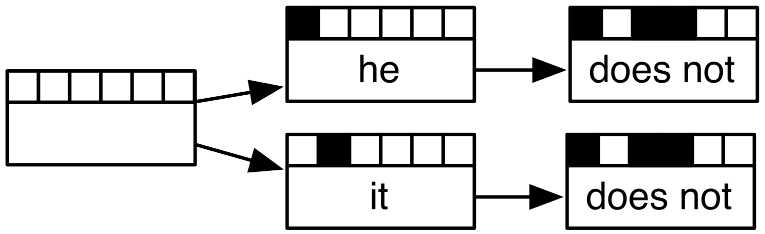


- Worse hypothesis is dropped

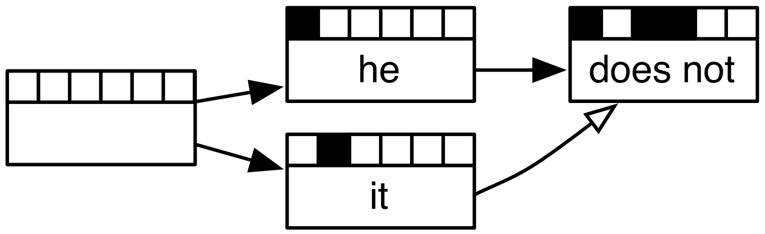


Recombination

- Two hypothesis paths lead to hypotheses indistinguishable in subsequent search
  - same number of foreign words translated
  - same last two English words in output (assuming trigram language model)
  - same last foreign word translated
  - different scores



- Worse hypothesis is dropped



Restrictions on Recombination

- **Translation model:** Phrase translation independent from each other
  $\rightarrow$ no restriction to hypothesis recombination

- **Language model:** Last $n-1$ words used as history in n-gram language model
  $\rightarrow$ recombined hypotheses must match in their last $n-1$ words

- **Reordering model:** Distance-based reordering model based on distance to end position of previous input phrase
  $\rightarrow$ recombined hypotheses must have that same end position

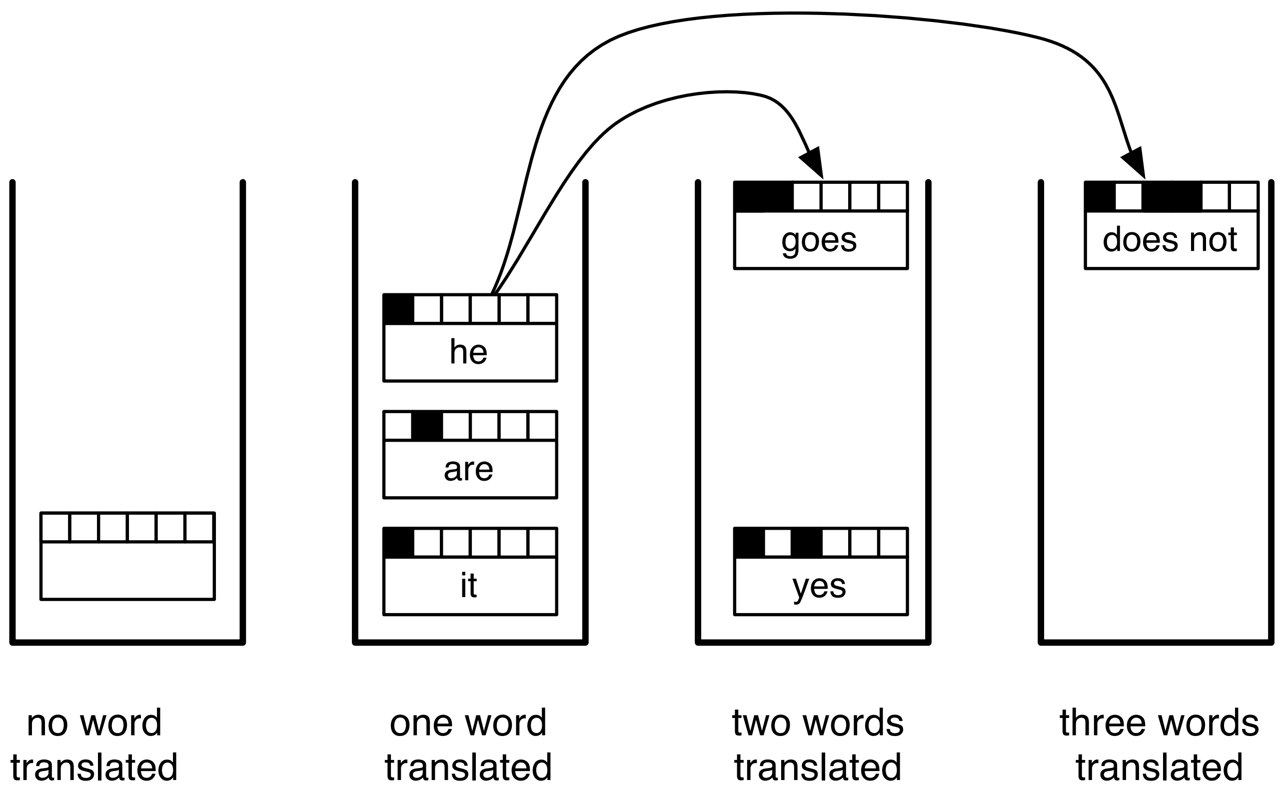
Pruning

- Recombination reduces search space, but not enough

  (we still have a NP complete problem on our hands)

- Pruning: remove bad hypotheses early
  - put comparable hypothesis into stacks
    (hypotheses that have translated same number of input words)
  - limit number of hypotheses in each stack

Stacks



- Hypothesis expansion in a stack decoder
  - translation option is applied to hypothesis
  - new hypothesis is dropped into a stack further down

Stack Decoding Algorithm

```
place empty hypothesis into stack 0
for all stacks 0...n − 1 do
   for all hypotheses in stack do
      for all translation options do
         if applicable then
            create new hypothesis
            place in stack
            recombine with existing hypothesis if possible
            prune stack if too big
         end if
      end for
   end for
end for
```

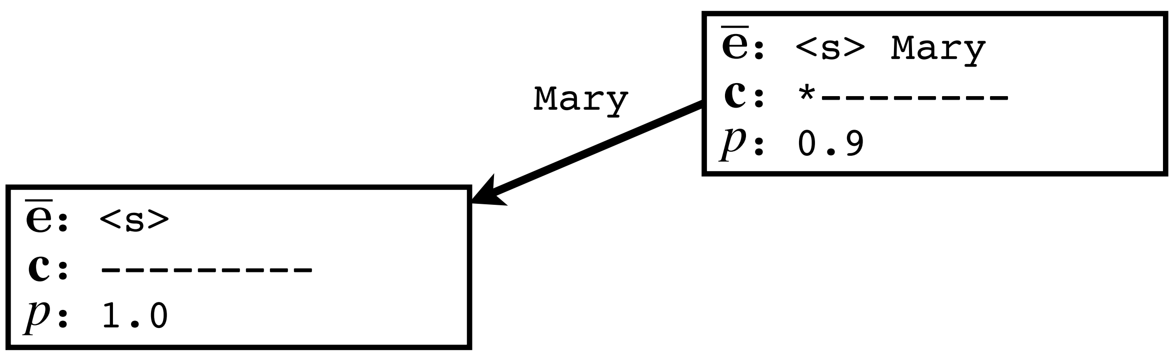
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...

$\bar{\mathbf{e}}$:	<s>
c:	---------
p:	1.0

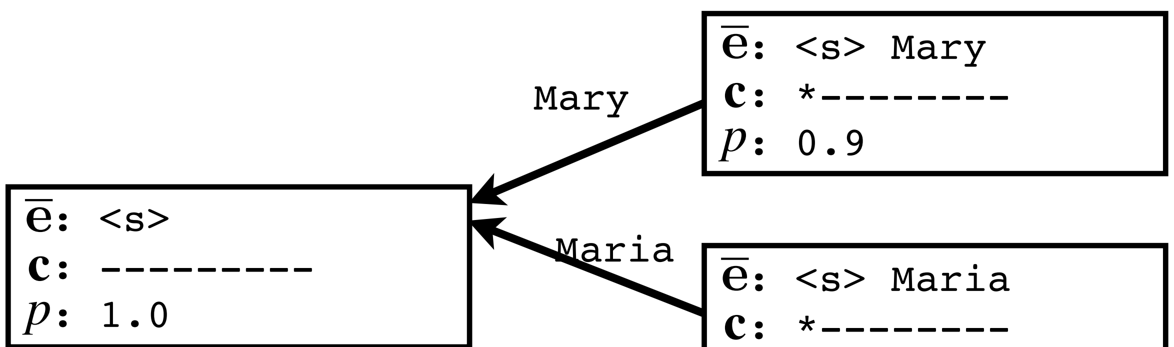
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



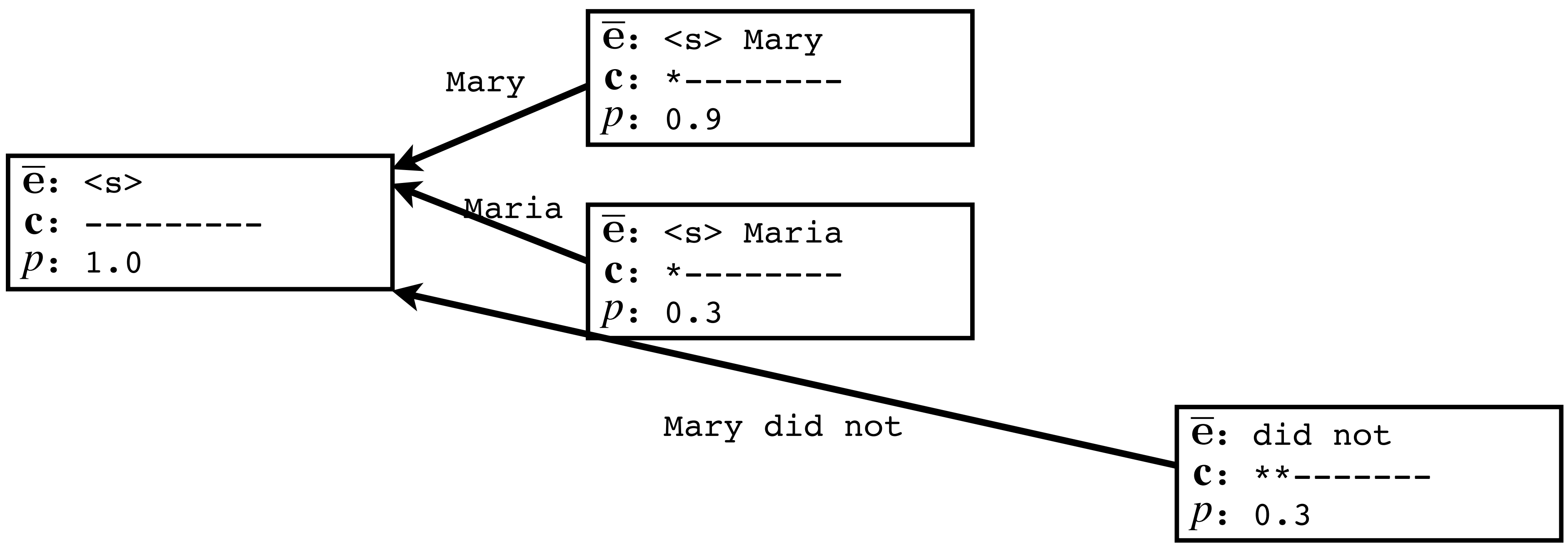
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



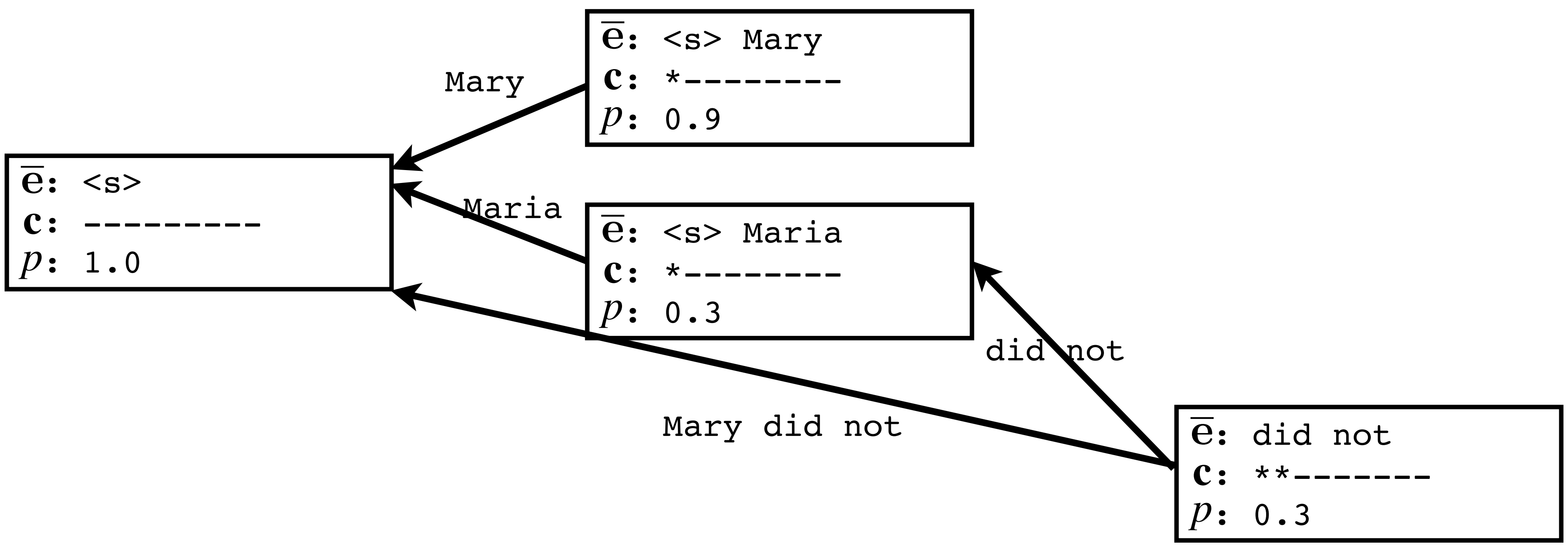
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



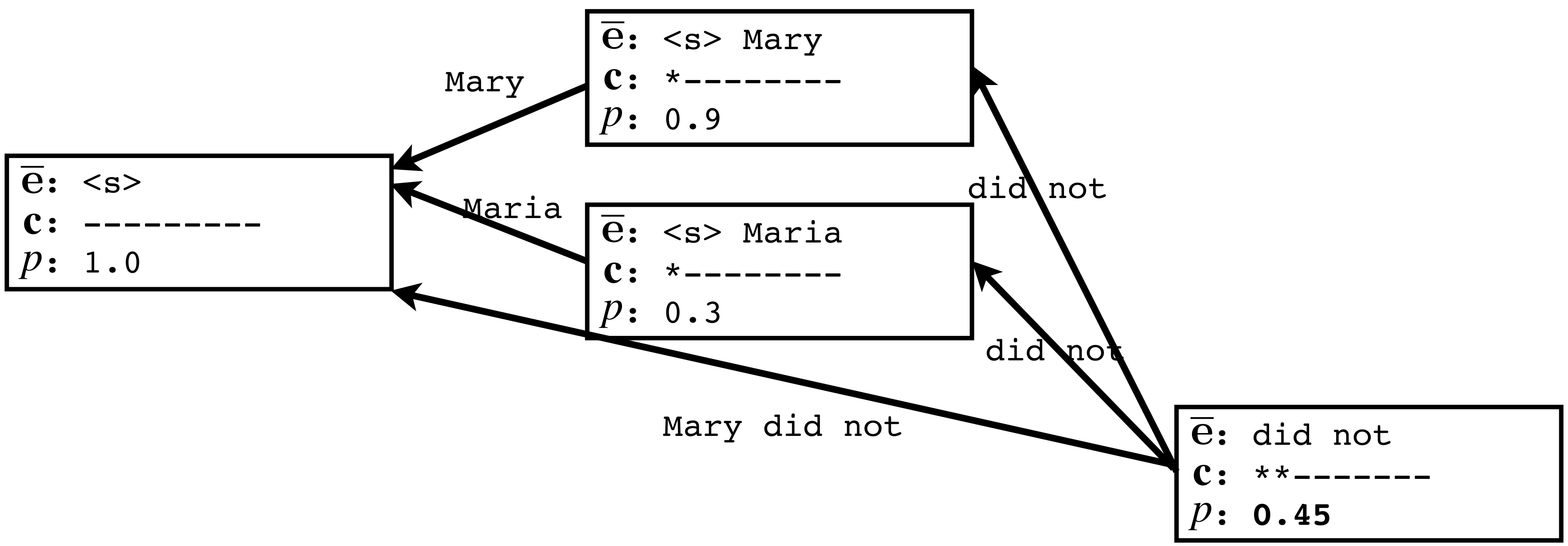
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



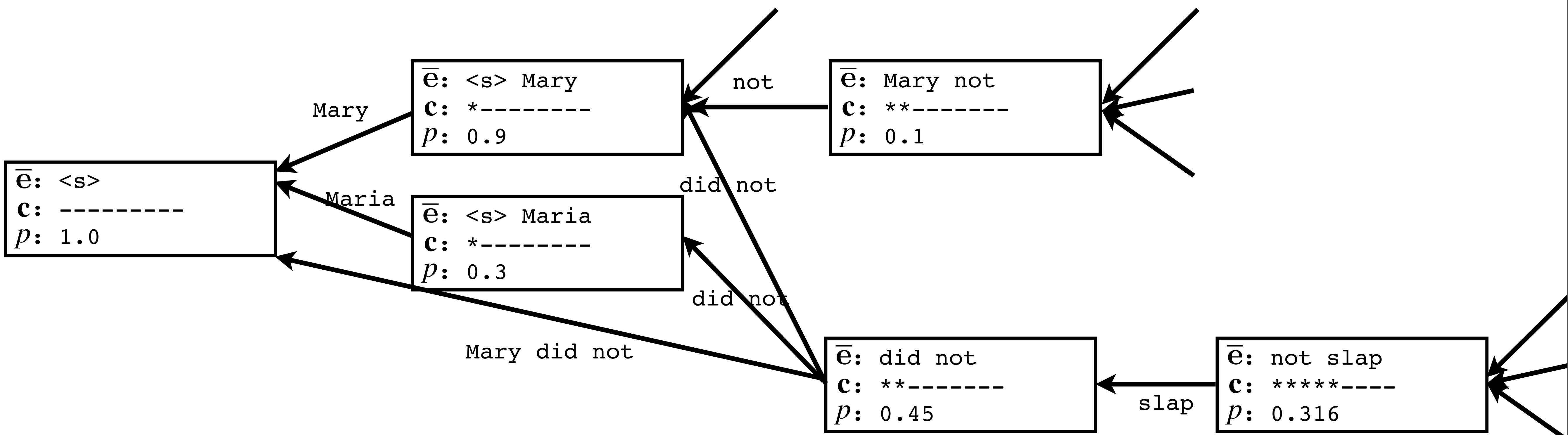
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



Pruning

- Pruning strategies
  - histogram pruning: keep at most k hypotheses in each stack
  - stack pruning: keep hypothesis with score $\alpha \times$ best score ($\alpha < 1$)

- Computational time complexity of decoding with histogram pruning

$$O(\text{max stack size} \times \text{translation options} \times \text{sentence length})$$

- Number of translation options is linear with sentence length, hence:

$$O(\text{max stack size} \times \text{sentence length}^2)$$

- Quadratic complexity

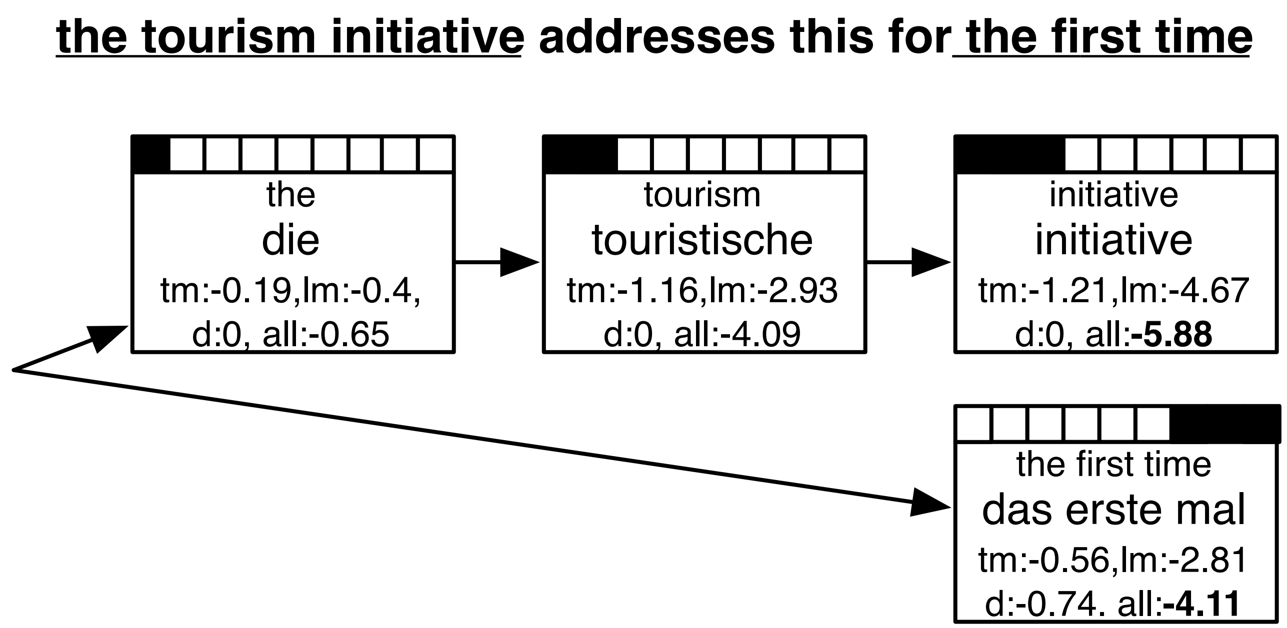
Reordering Limits

- Limiting reordering to maximum reordering distance
- Typical reordering distance 5–8 words
  - depending on language pair
  - larger reordering limit hurts translation quality
- Reduces complexity to linear

$$O(\text{max stack size} \times \text{sentence length})$$

- Speed / quality trade-off by setting maximum stack size

Translating the Easy Part First?

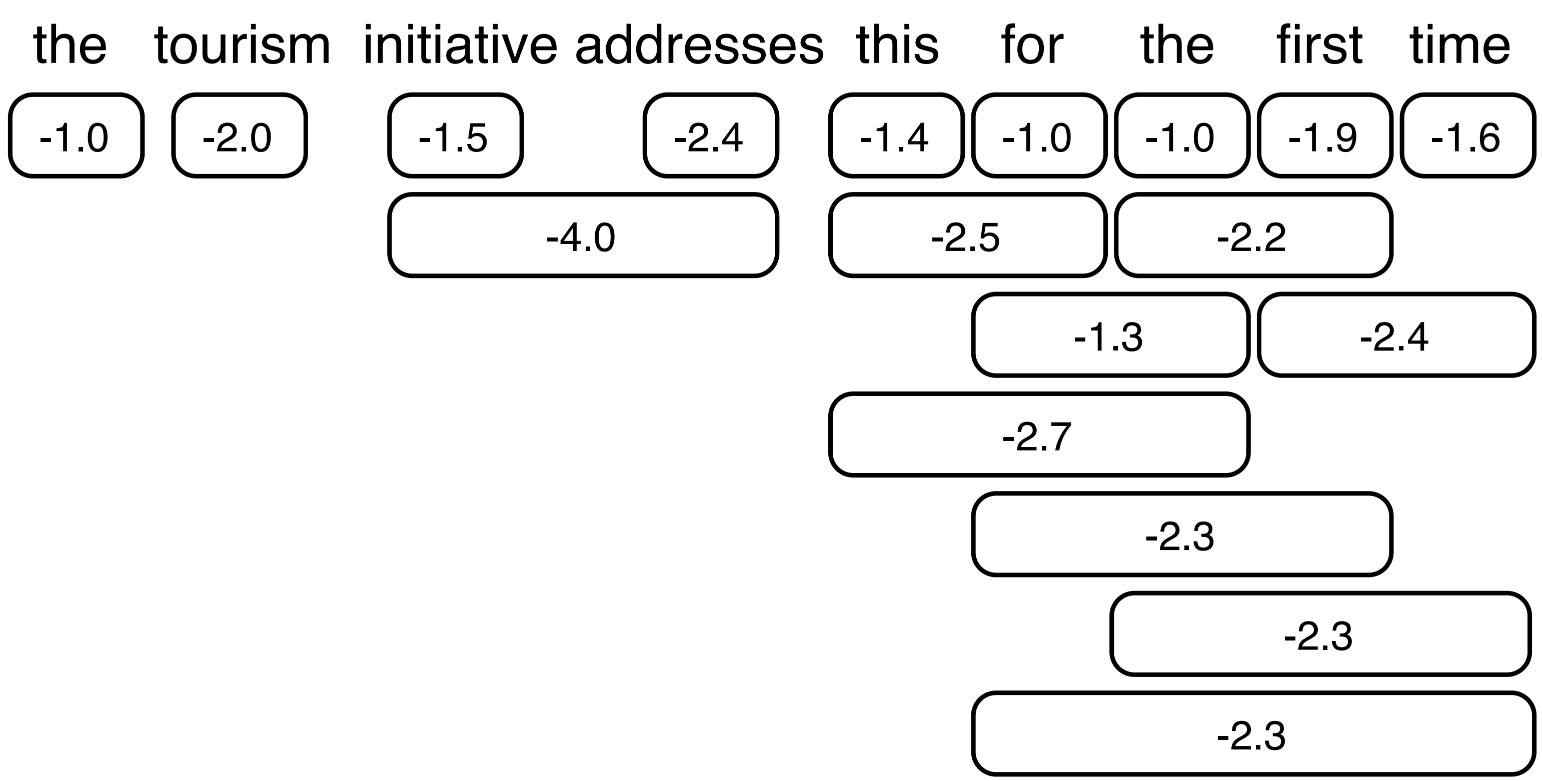


both hypotheses translate 3 words
worse hypothesis has better score

Estimating Future Cost

- Future cost estimate: how expensive is translation of rest of sentence?
- Optimistic: choose cheapest translation options
- Cost for each translation option
  - **translation model**: cost known
  - **language model:** output words known, but not context
    $\rightarrow$ estimate without context
  - **reordering model:** unknown, ignored for future cost estimation

Cost Estimates from Translation Options



cost of cheapest translation options for each input span (log-probabilities)

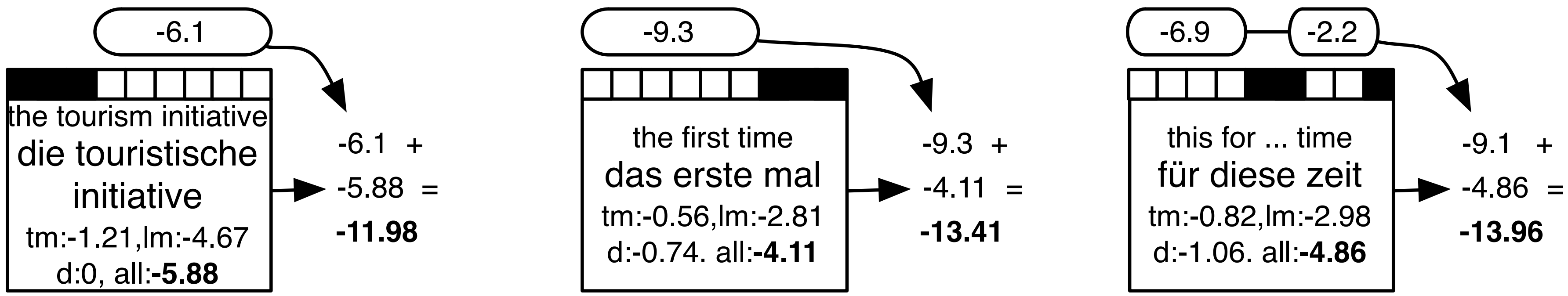
Cost Estimates for all Spans

- Compute cost estimate for all contiguous spans by combining cheapest options

first word	future cost estimate for n words (from first)								
	1	2	3	4	5	6	7	8	9
the	-1.0	-3.0	-4.5	-6.9	-8.3	-9.3	-9.6	-10.6	-10.6
tourism	-2.0	-3.5	-5.9	-7.3	-8.3	-8.6	-9.6	-9.6	
initiative	-1.5	-3.9	-5.3	-6.3	-6.6	-7.6	-7.6		
addresses	-2.4	-3.8	-4.8	-5.1	-6.1	-6.1			
this	-1.4	-2.4	-2.7	-3.7	-3.7				
for	-1.0	-1.3	-2.3	-2.3					
the	-1.0	-2.2	-2.3						
first	-1.9	-2.4							
time	-1.6								

- Function words cheaper (the: -1.0) than content words (tourism -2.0)
- Common phrases cheaper (for the first time: -2.3)
  than unusual ones (tourism initiative addresses: -5.9)

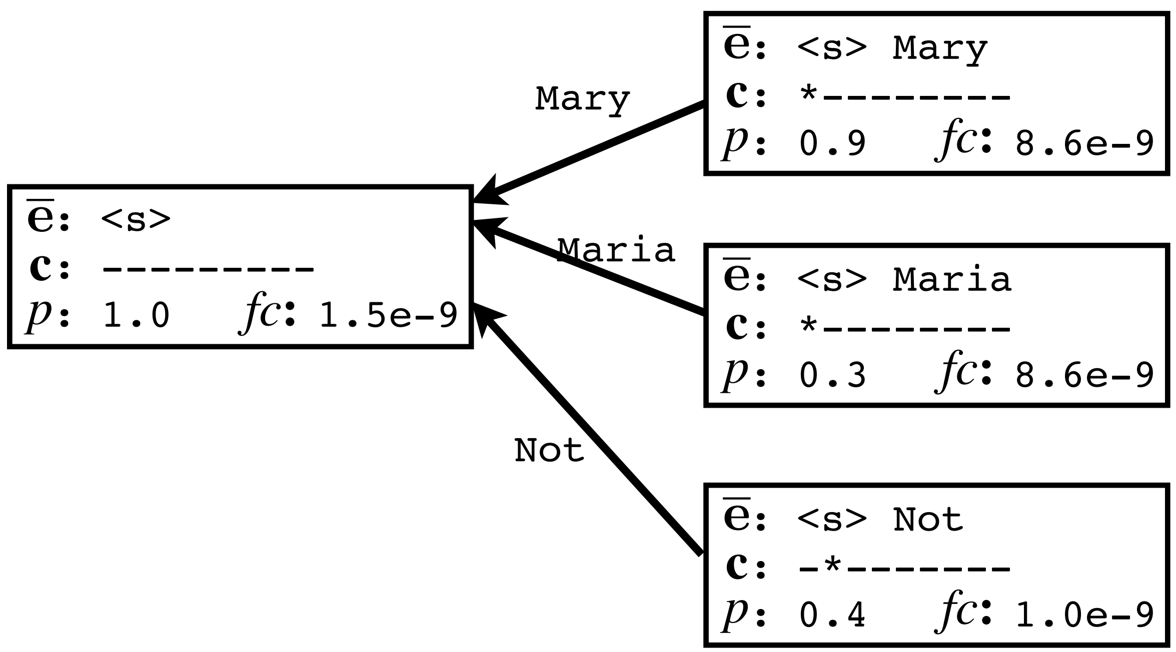
Combining Score and Future Cost



- Hypothesis score and future cost estimate are combined for pruning
  - left hypothesis starts with hard part: the tourism initiative
    score: -5.88, future cost: -6.1 → total cost -11.98
  - middle hypothesis starts with easiest part: the first time
    score: -4.11, future cost: -9.3 → total cost -13.41
  - right hypothesis picks easy parts: this for ... time
    score: -4.86, future cost: -9.1 → total cost -13.96

f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...

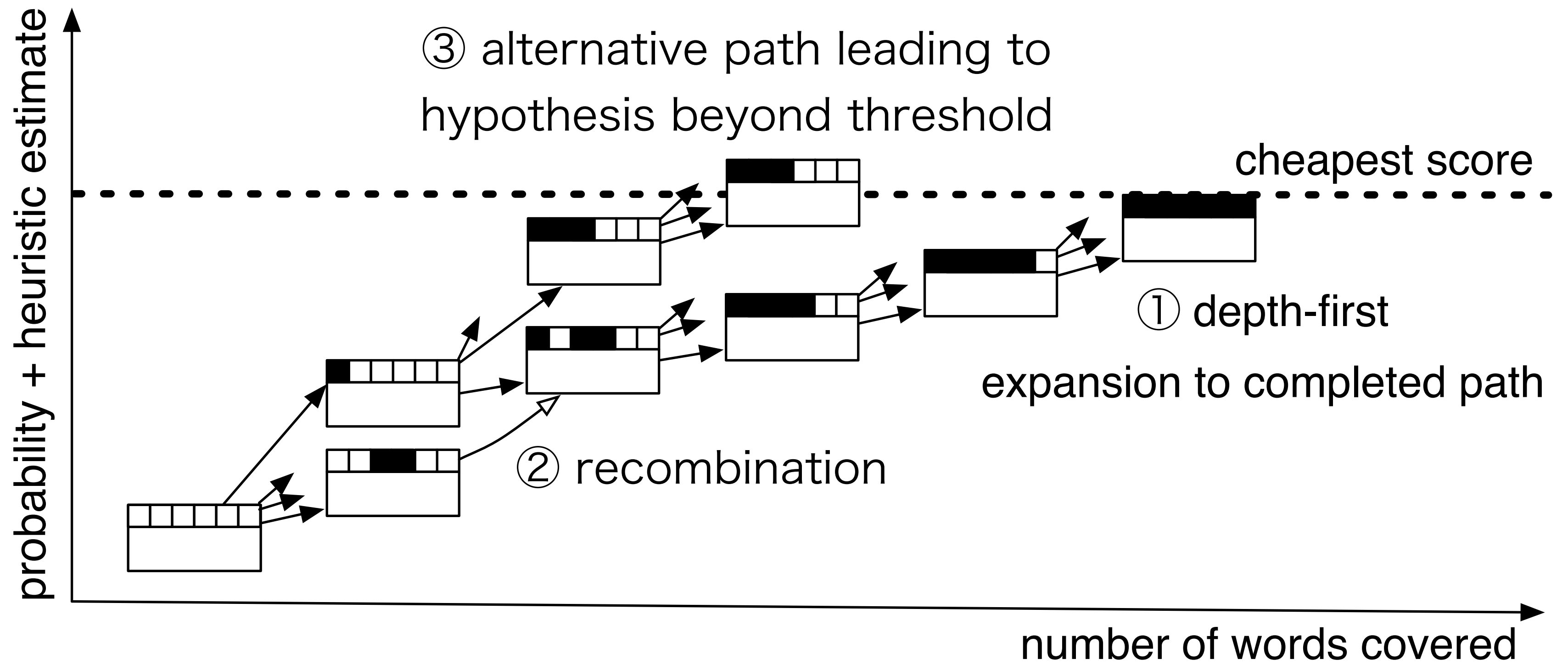


Future costs make these hypotheses comparable.

Other Decoding Algorithms

- A* search
- Greedy hill-climbing
- Using finite state transducers (standard toolkits)

A* Search



- Uses *admissible* future cost heuristic: never overestimates cost
- Translation agenda: create hypothesis with lowest score + heuristic cost
- Done, when complete hypothesis created

Greedy Hill-Climbing

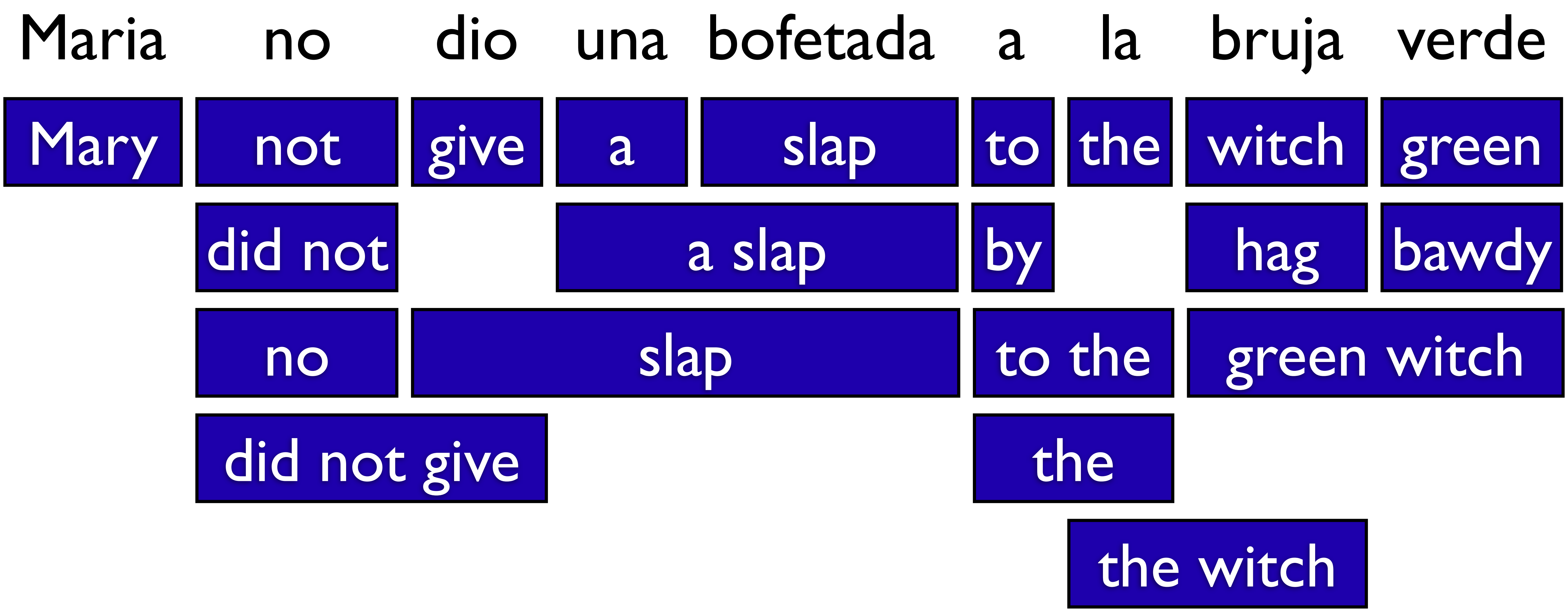
- Create one complete hypothesis with depth-first search (or other means)
- Search for better hypotheses by applying change operators
  - change the translation of a word or phrase
  - combine the translation of two words into a phrase
  - split up the translation of a phrase into two smaller phrase translations
  - move parts of the output into a different position
  - swap parts of the output with the output at a different part of the sentence
- Terminates if no operator application produces a better translation

Marginal Decoding

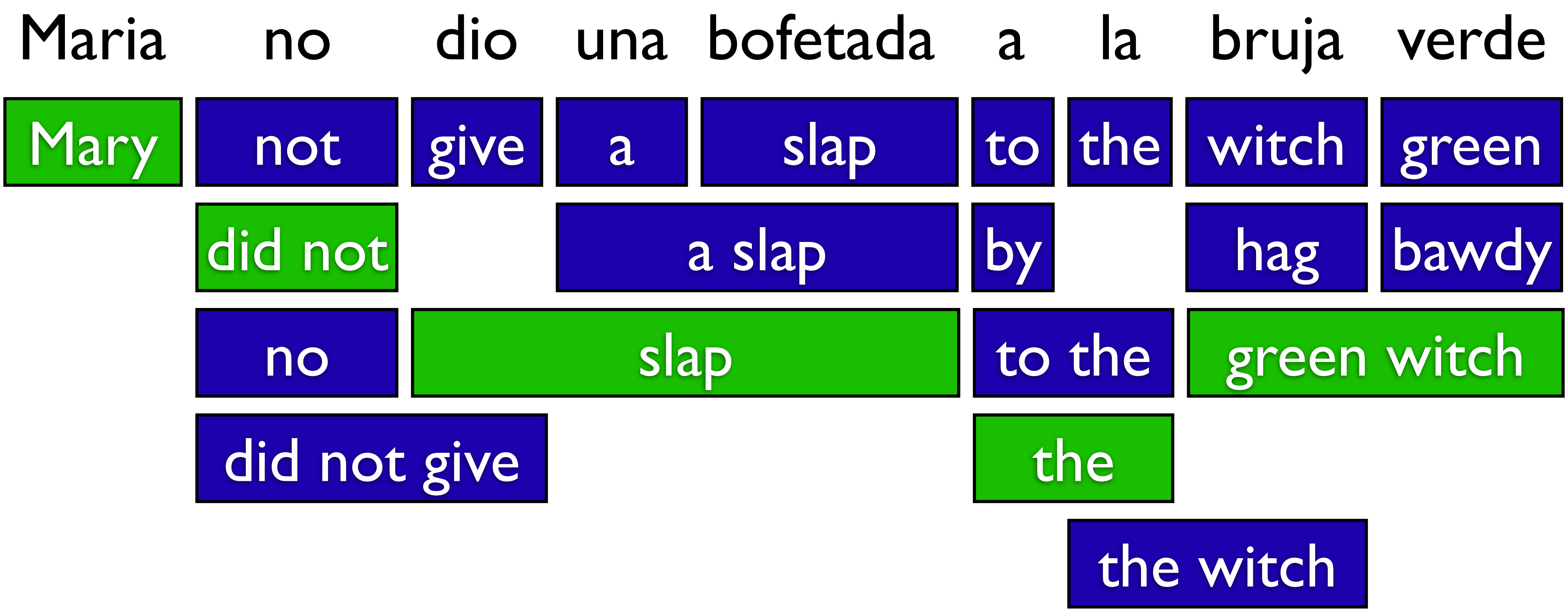
$$
\begin{aligned}
\mathbf{e}^* &= \arg\max_{\mathbf{e}} p(\mathbf{e} \mid \mathbf{f}) \\
&= \arg\max_{\mathbf{e}} p(\mathbf{f} \mid \mathbf{e}) \times p(\mathbf{e}) \\
&\approx \arg\max_{\mathbf{e}} p(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \times p(\mathbf{e})
\end{aligned}
$$

Does this last approximation matter?

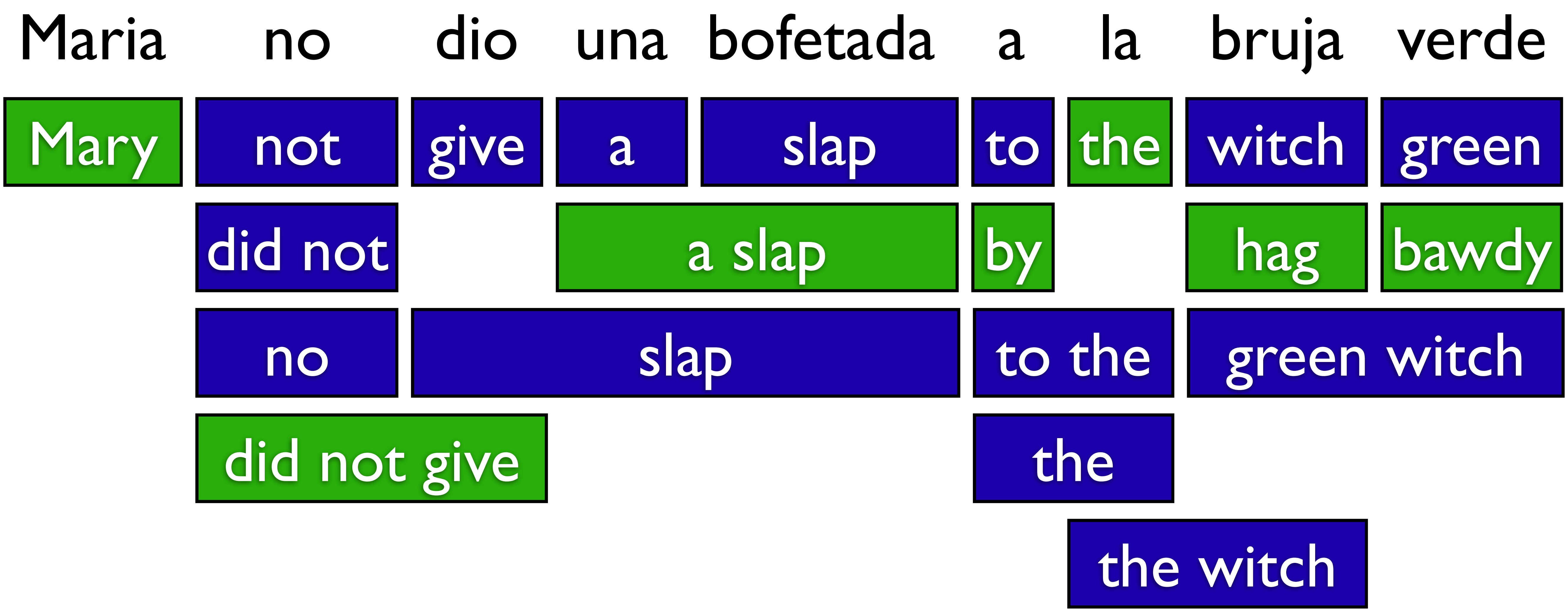
- Variational & MCMC explored
- <u>*marginal*</u> benefits, depending on training
- Really hard problem (Sima'an, 1997)



Adapted from Koehn (2006)



Adapted from Koehn (2006)



Adapted from Koehn (2006)

Decoding algorithm

- Translation as a search problem
- Partial hypothesis keeps track of
  - which source words have been translated (*coverage vector*)
  - *n*-1 most recent words of English (for LM!)
  - a *back pointer* list to the previous hypothesis + (e,f) phrase pair used
  - the (partial) translation probability
  - the *estimated probability* of translating the remaining words (precomputed, a function of the coverage vector)
- **Start state**: no translated words, E=<s>, bp=nil
- **Goal state**: all translated words

Decoding algorithm

- Q[0] ← Start state
- for i = 0 to |**f**|-1
  - Keep *b* best hypotheses at Q[i]
  - for each hypothesis h in Q[i]
    - for each untranslated span in h.**c** for which there is a translation <e,f> in the phrase table
      - h' = h extend by <e,f>
      - Is there an item in Q[|h'.**c**|] with = LM state?
        - yes: update the item bp list and probability
        - no: Q[|h'.**c**|] ← h'
- Find the best hypothesis in Q[|**f**|], reconstruction translation by following back pointers

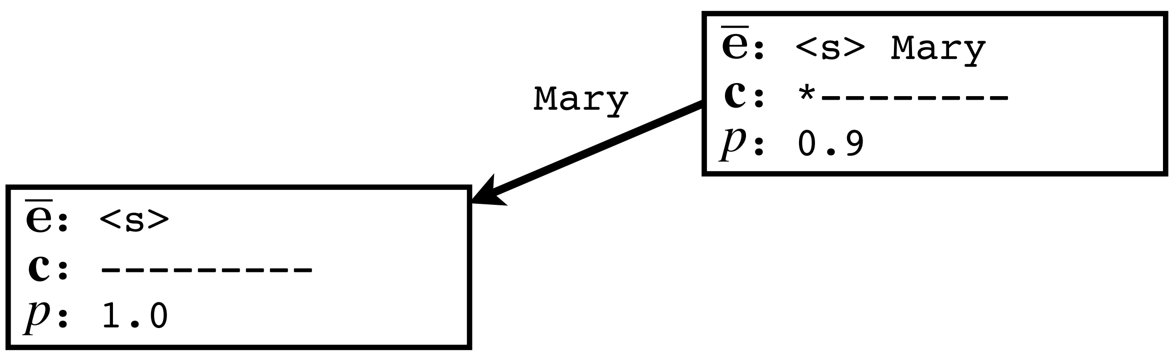
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...

$\bar{\mathbf{e}}$: <s>
$\mathbf{c}$: ---------
p: 1.0

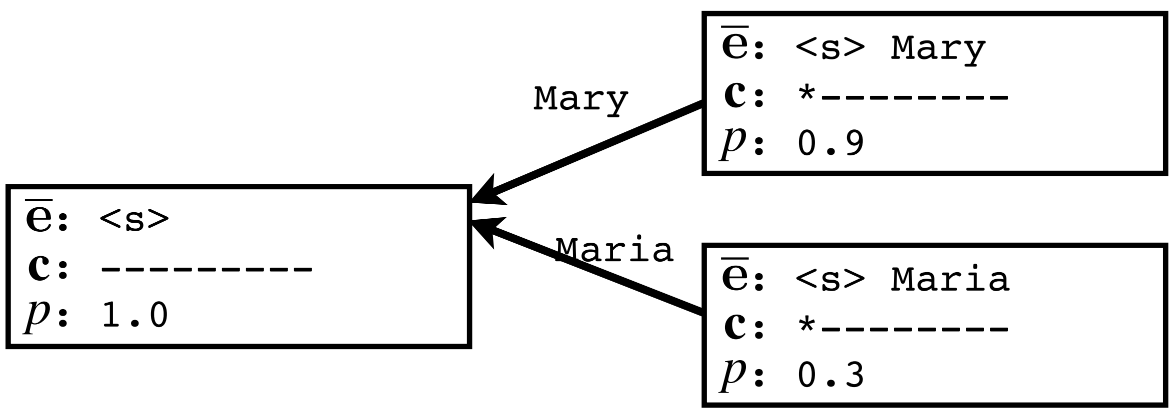
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



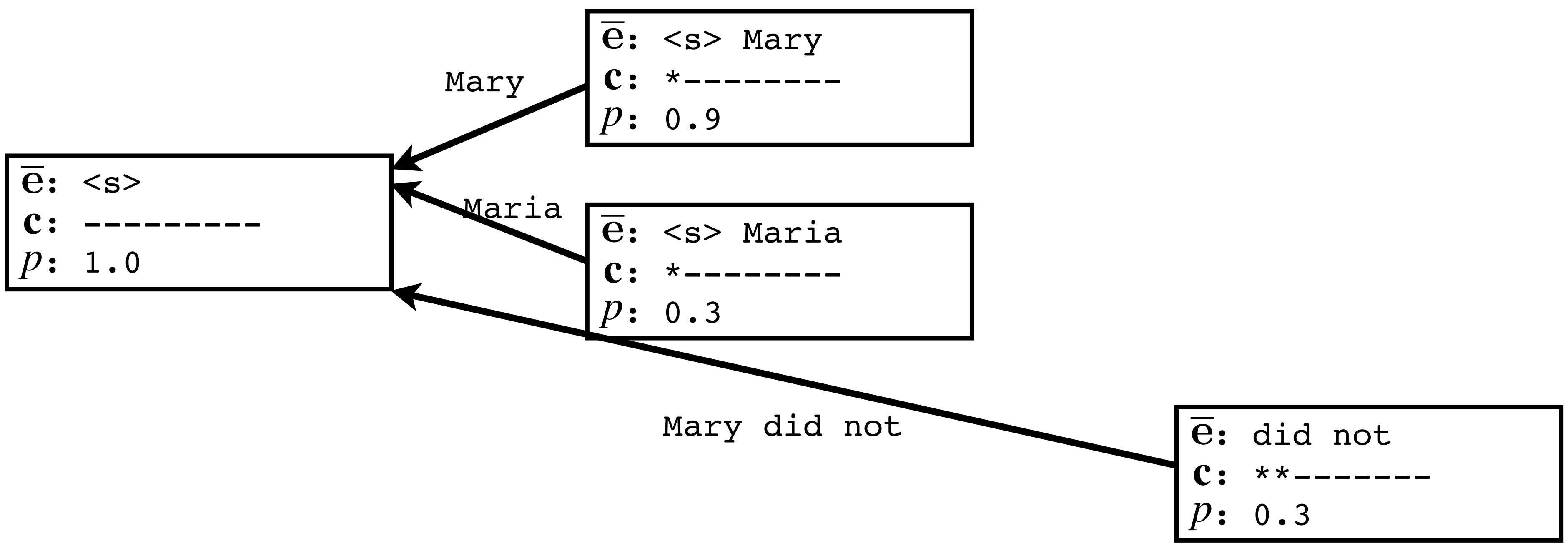
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



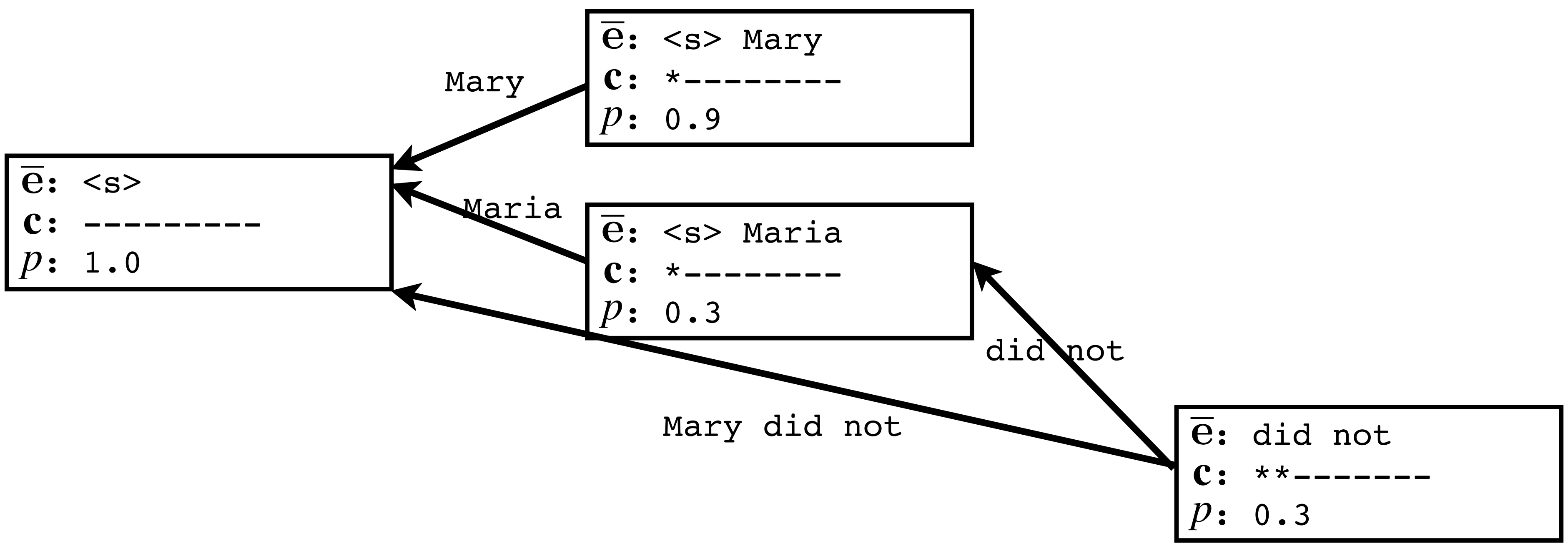
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



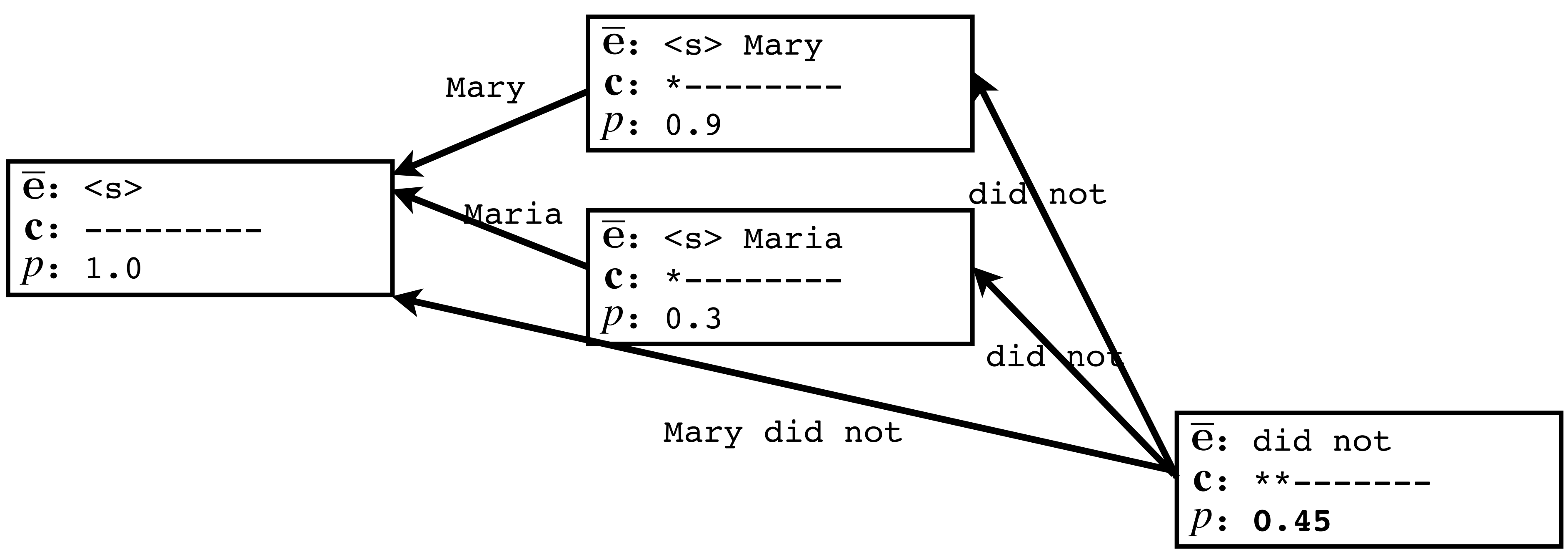
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



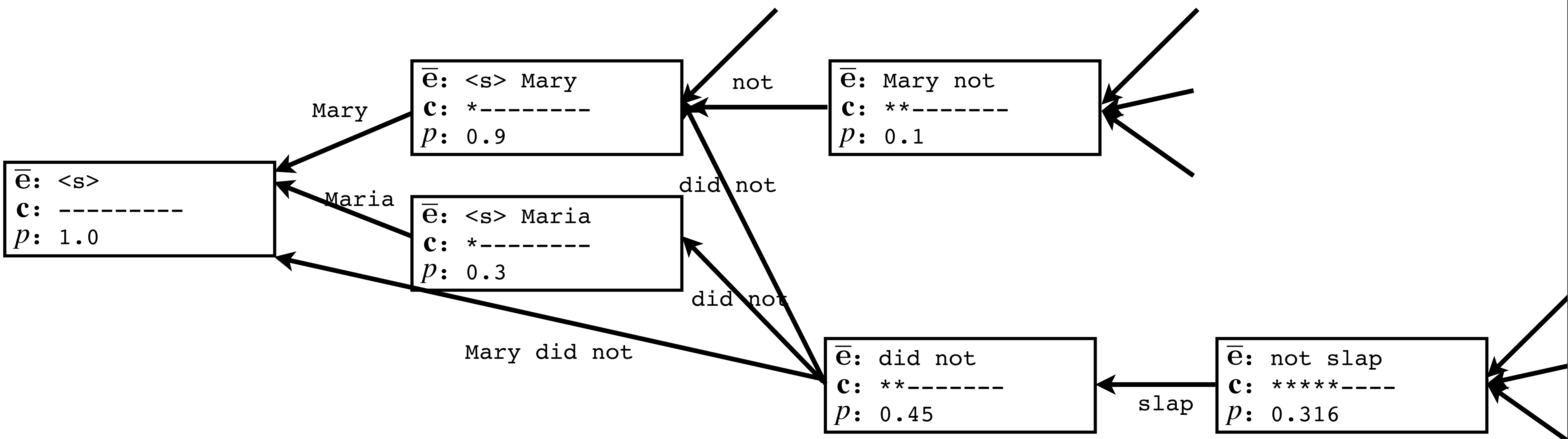
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...

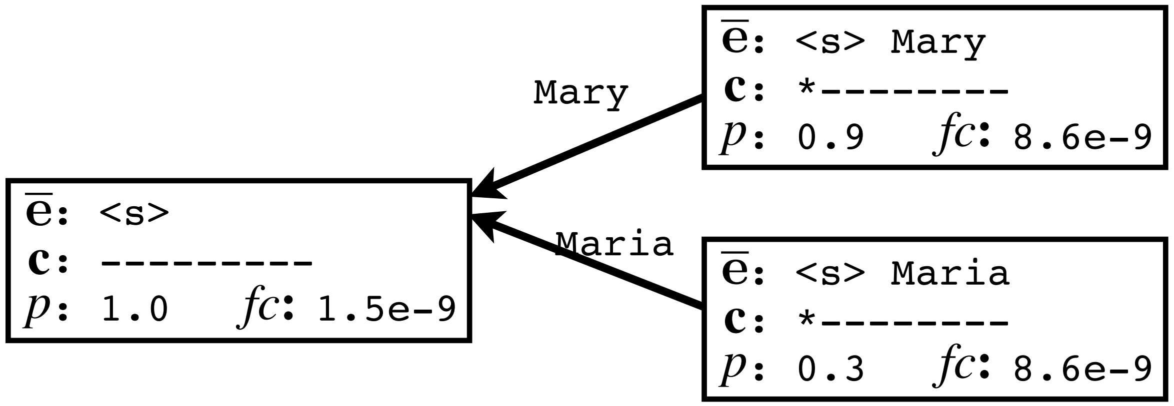


Reordering

- Language express words in different orders
  - bruja verde *vs.* green witch
- Phrase pairs can "memorize" some of these
- More general: in decoding, "skip ahead"
- Problem:
  - Won't "easy parts" of the sentence be translated first?
- Solution:
  - **Future cost estimate**
  - For every **coverage vector**, estimate what it will cost to translate the remaining untranslated words
  - When pruning, use p * future cost!

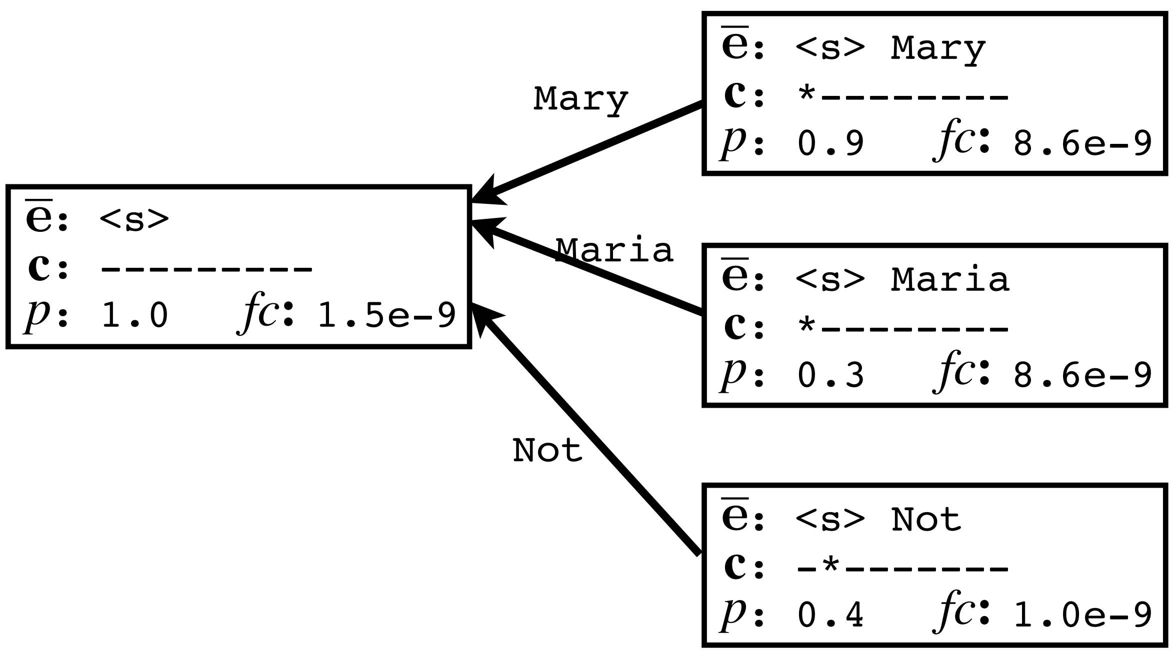
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



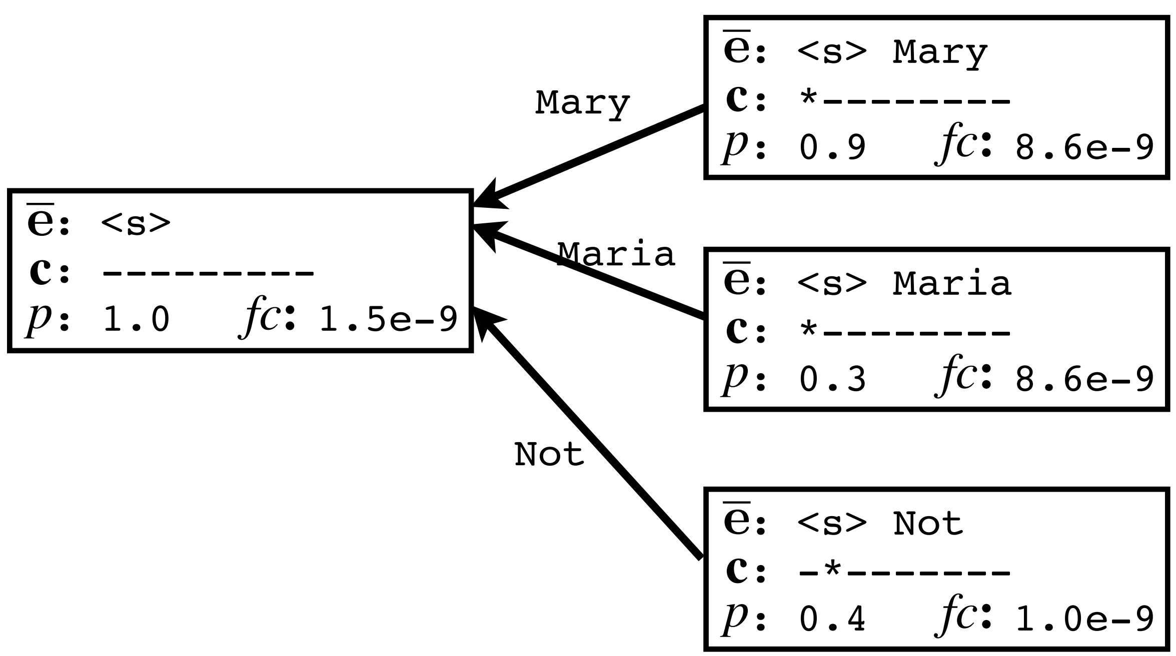
f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



f: Maria no dio una bofetada a la bruja verde

Q[0] Q[1] Q[2] ...



Future costs make these hypotheses comparable.

Decoding summary

- Finding the best hypothesis is NP-hard
  - Even with no language model, there are an exponential number of states!
  - Solution 1: limit reordering
  - Solution 2: (lossy) pruning