Statistical Machine Translation

-decoding-

Artem Sokolov
Computerlinguistik
Universität Heidelberg
Sommersemester 2015

material from P. Koehn

$$e_{\mathsf{best}} = \arg\max_e p(e|f) = \arg\max_e w \cdot \phi(e, f)$$

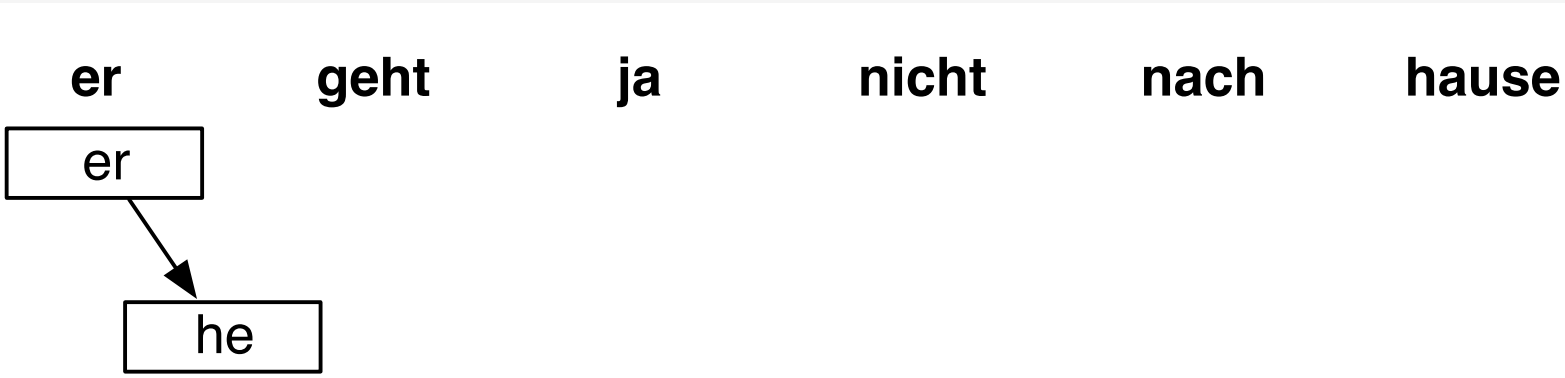
$$e_{\text{best}} = \arg\max_e p(e|f) = \arg\max_e w \cdot \phi(e, f)$$

$$e_{\text{best}} = \arg\max_e \prod_{i=1}^{I} (\phi(\bar{f}_i|\bar{e}_i) d(\text{start}_i - \text{end}_{i-1} - 1)) \prod_{j=1}^{|e|} p_{\text{LM}}(e_j|e_1, \ldots, e_{j-1})$$

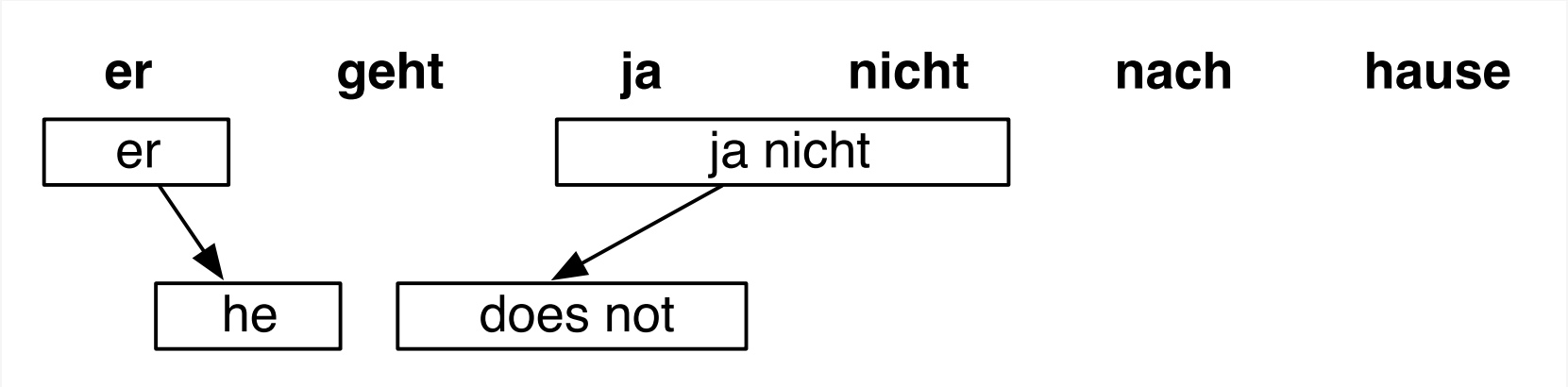
‘Almost Human’ Decoding

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

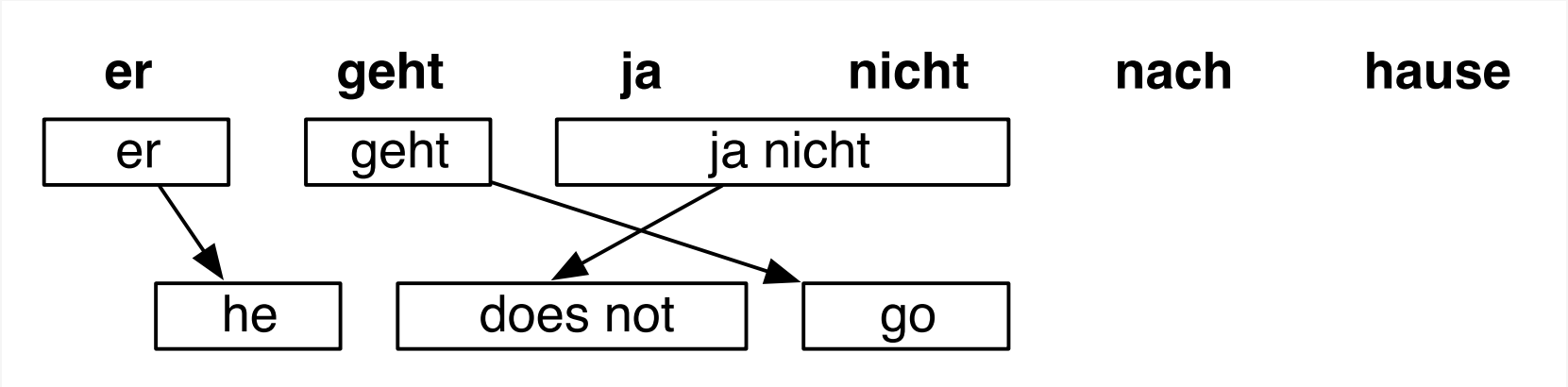
‘Almost Human’ Decoding



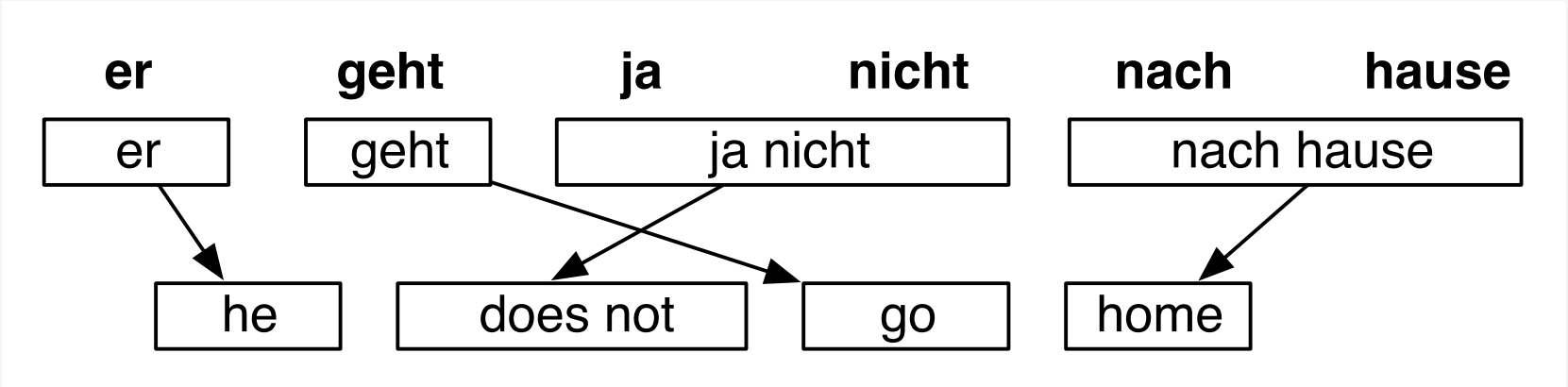
‘Almost Human’ Decoding



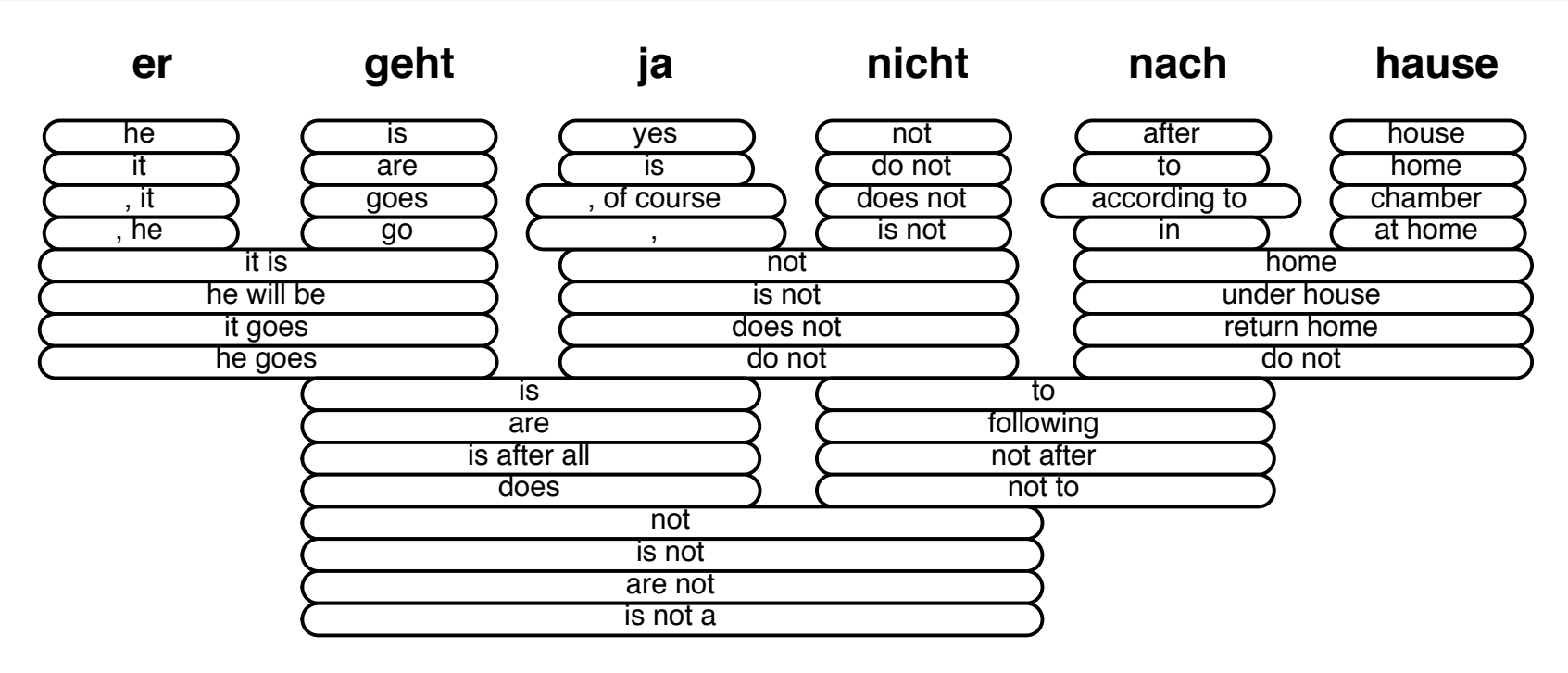
‘Almost Human’ Decoding



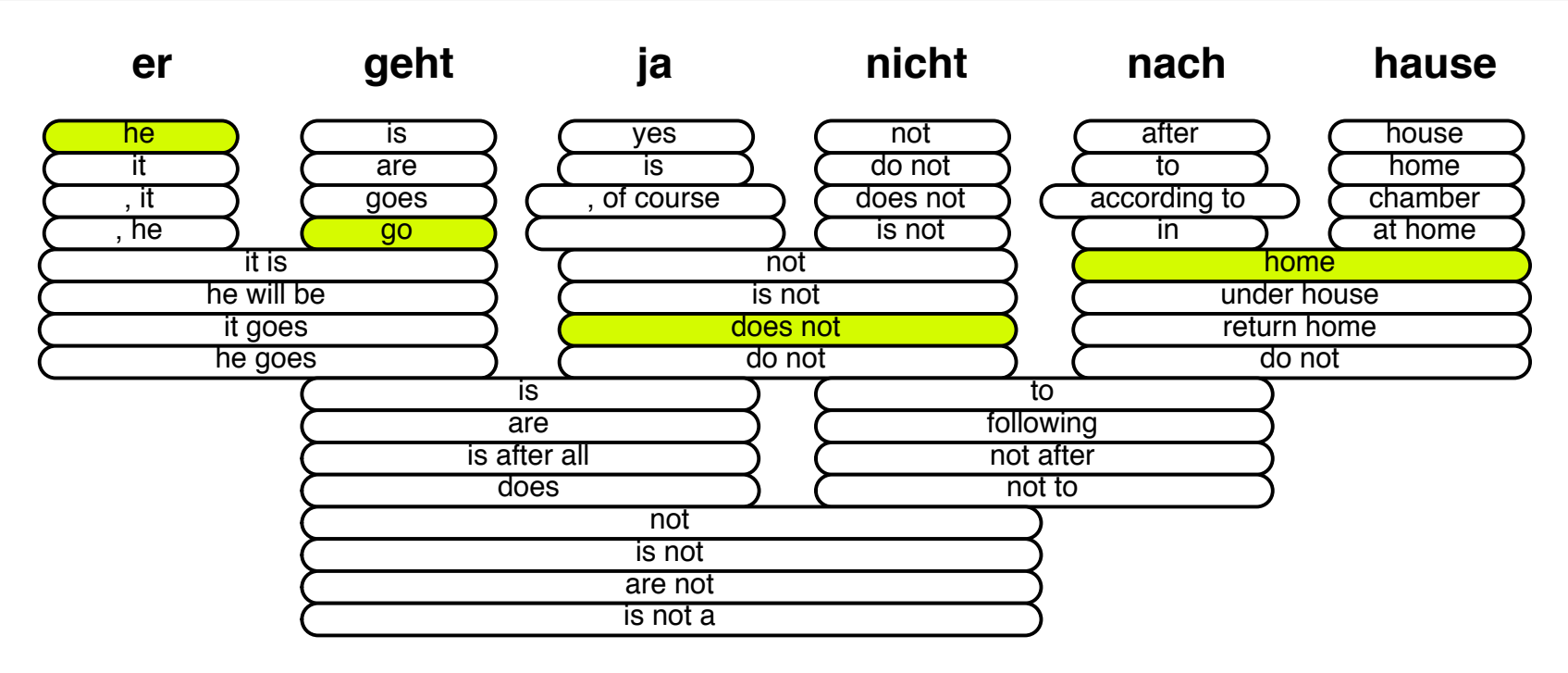
‘Almost Human’ Decoding



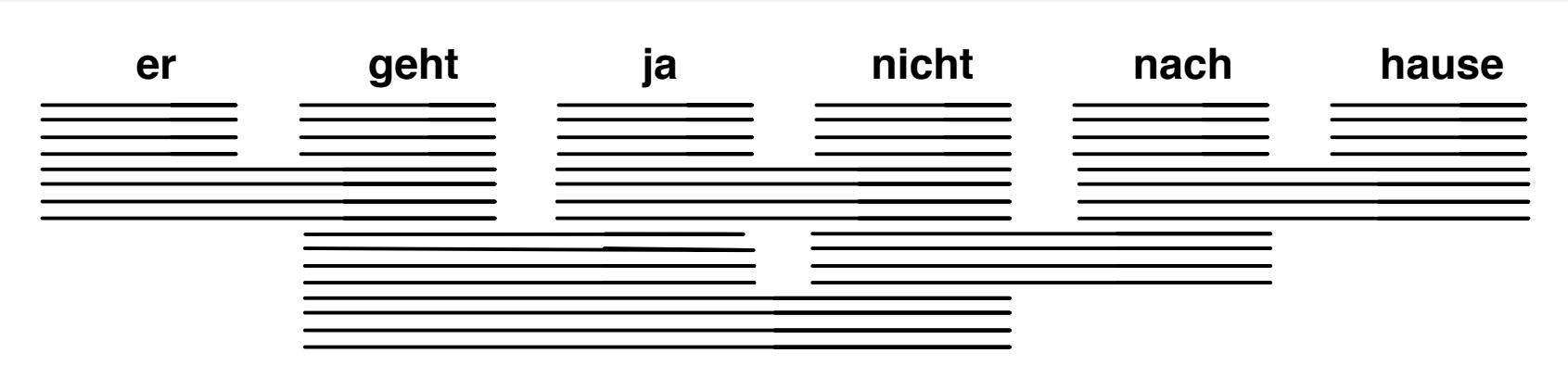
Decoding by Hypothesis Expansion



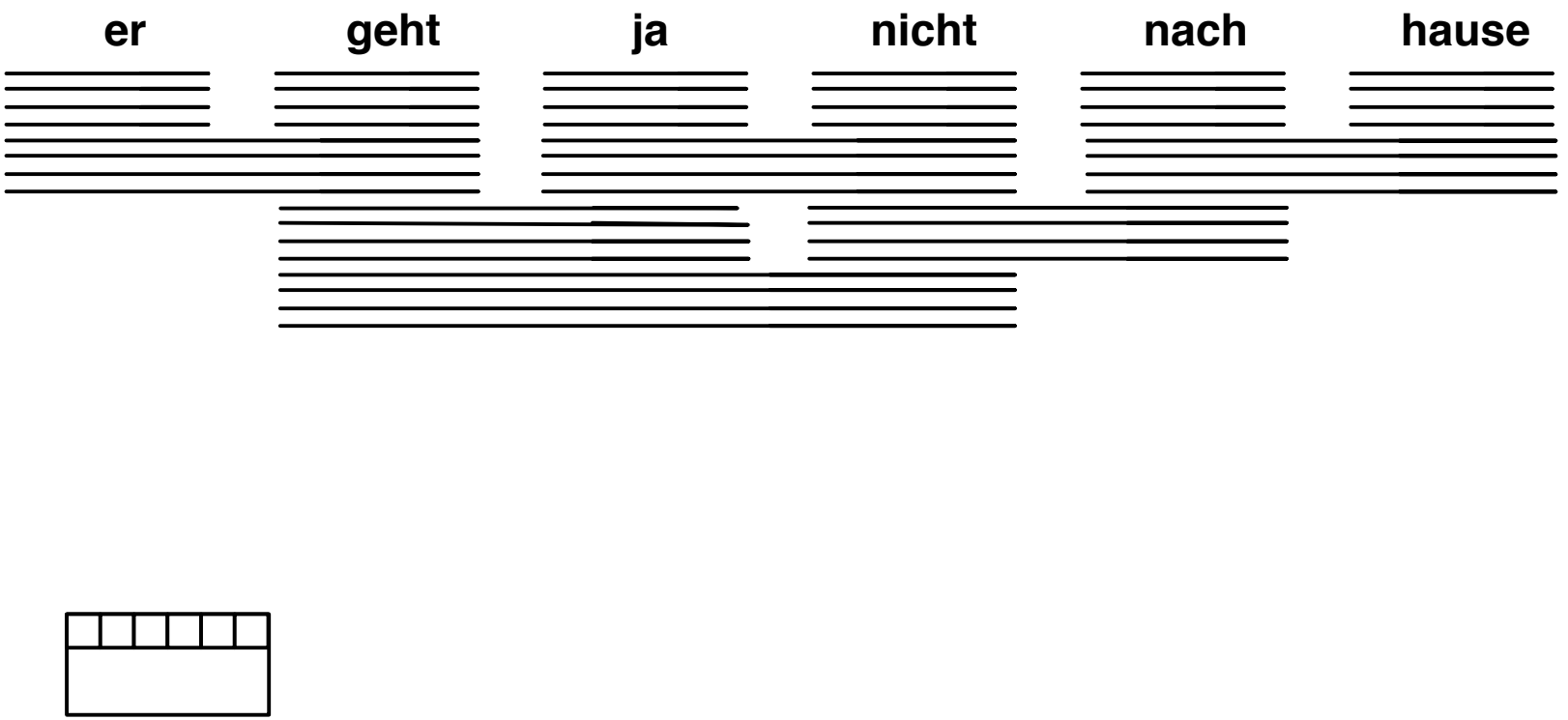
Decoding by Hypothesis Expansion



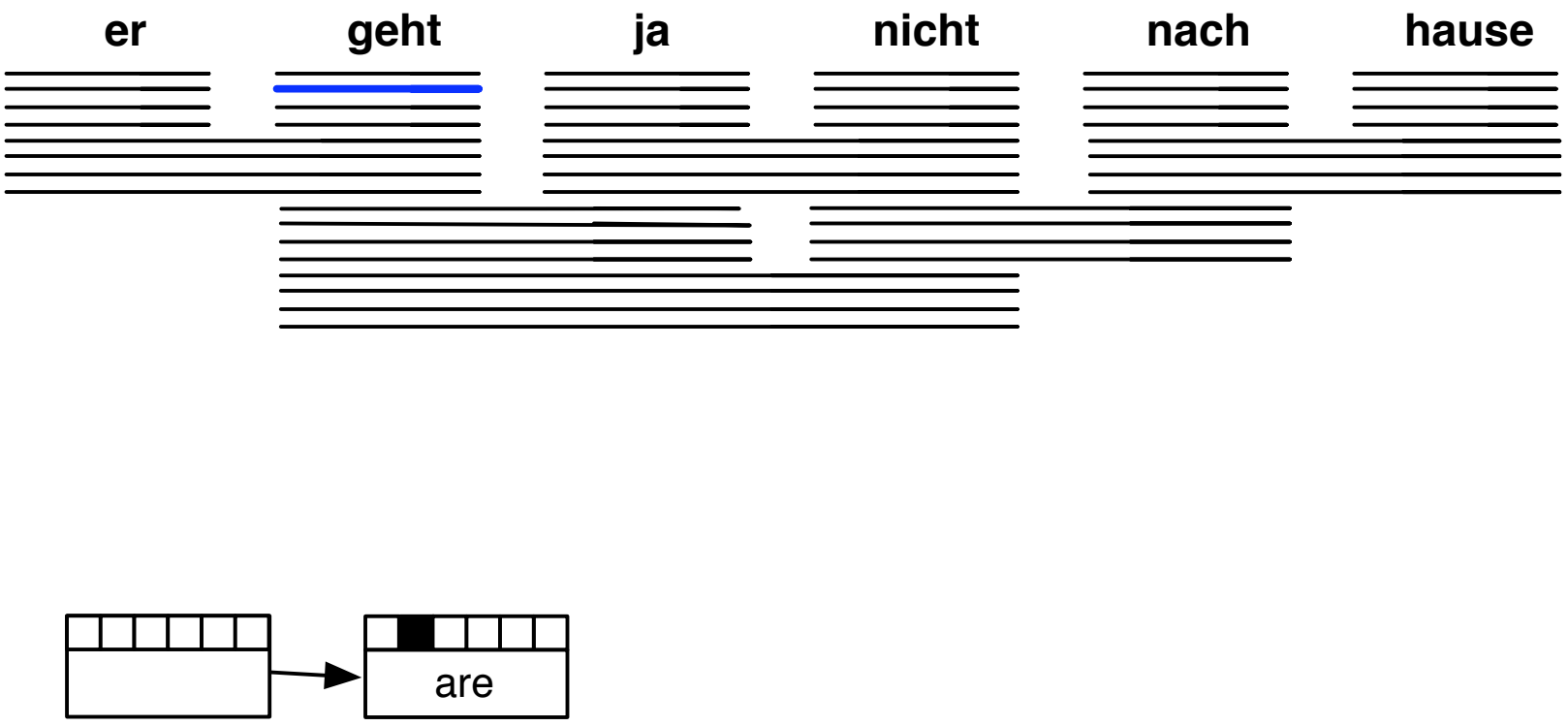
Consult phrase translation table for all possible input phrases, precompute **translation options** as all applicable phrase translations:



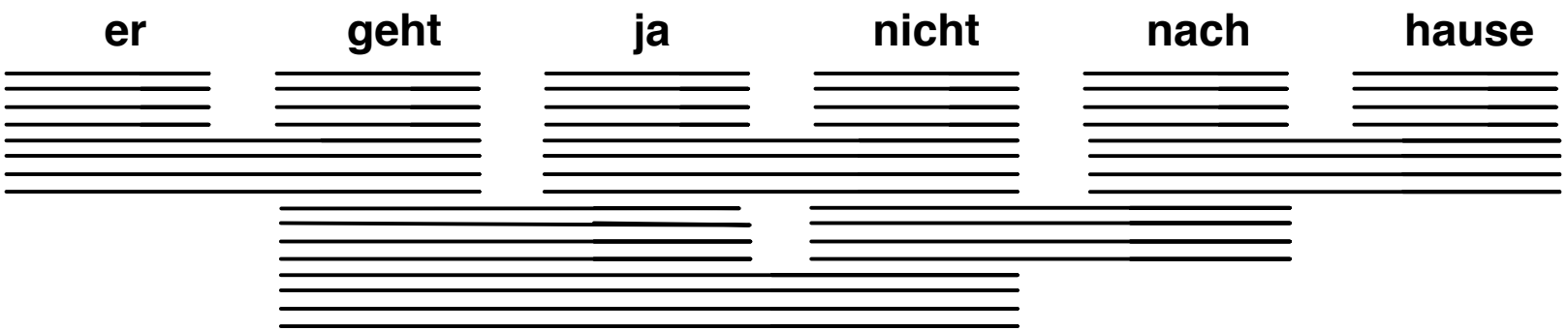
Initial hypothesis: No input phrase covered, no output produced:

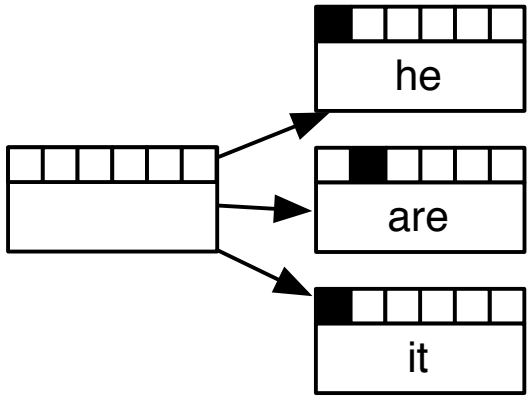


Hypothesis expansion: Pick translation option, create new hypothesis by constructing partial translation, mark off input:

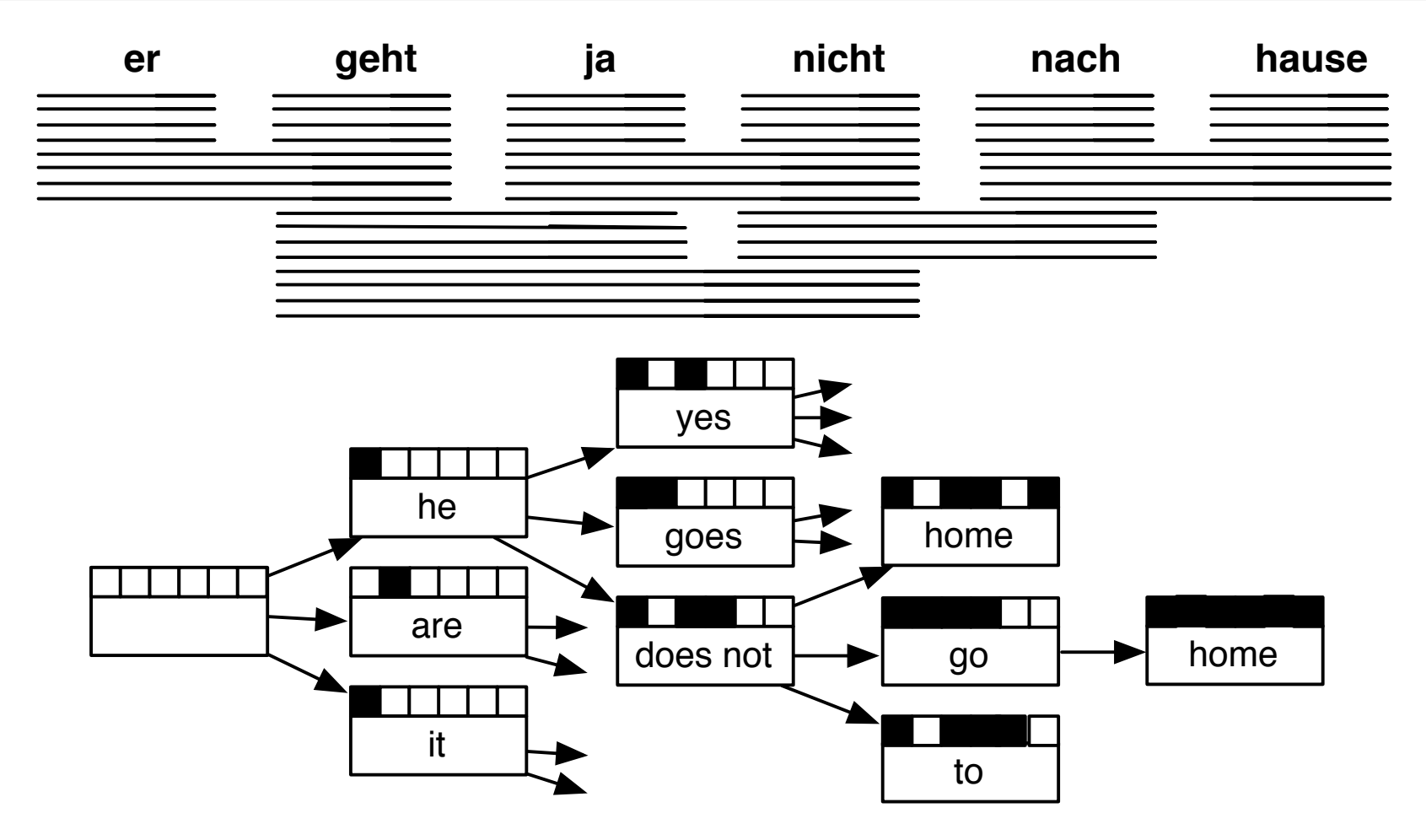


Create hypotheses for all other translation options:

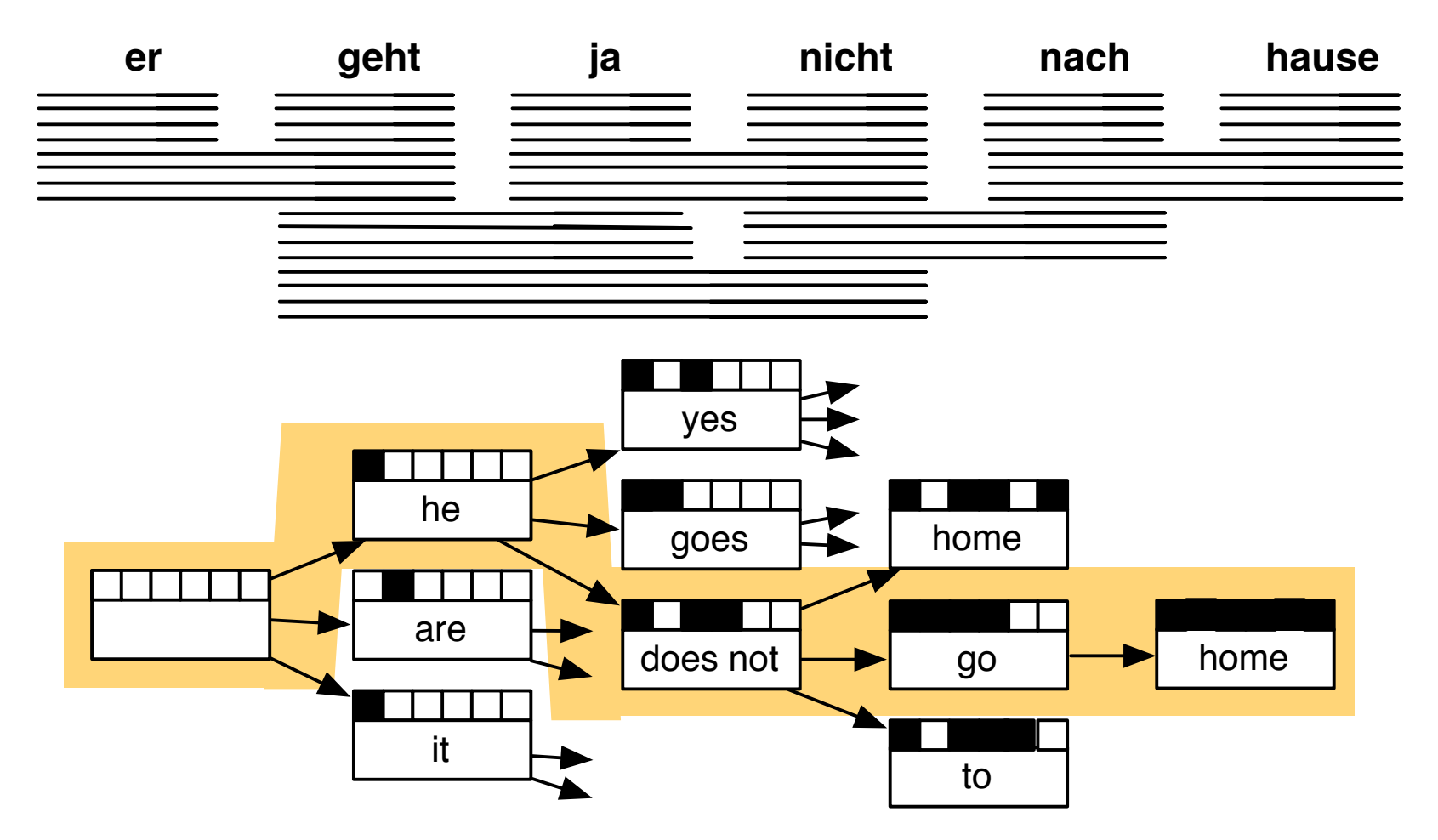




Create hypotheses from already created partial hypotheses:



Find best path by backtracking from highest scoring complete hypothesis:

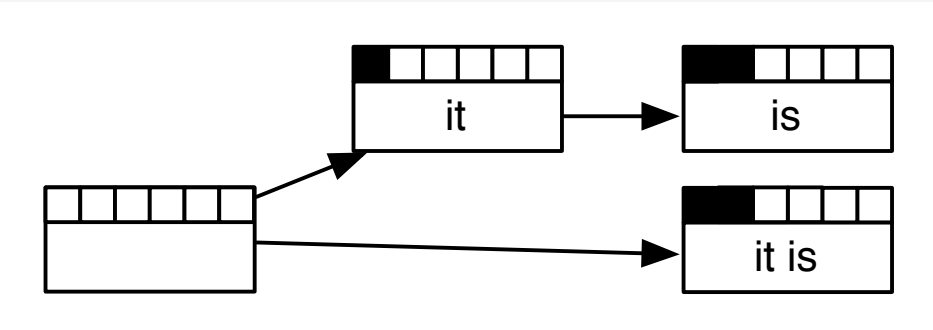


$$O(\sum_{i=1}^{\text{sentence length}} |\text{translation options}|^i) = O(|\text{translation options}|^{\text{sentence length}})$$

- machine translation decoding is NP-complete
- reduction of search space:
  - recombination (risk-free)
  - pruning (risky)

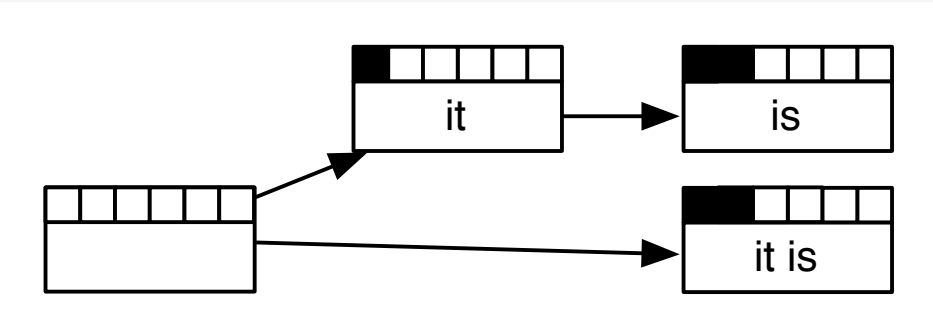
case 1:

- the same number of foreign words translated,
- the same English words in output:

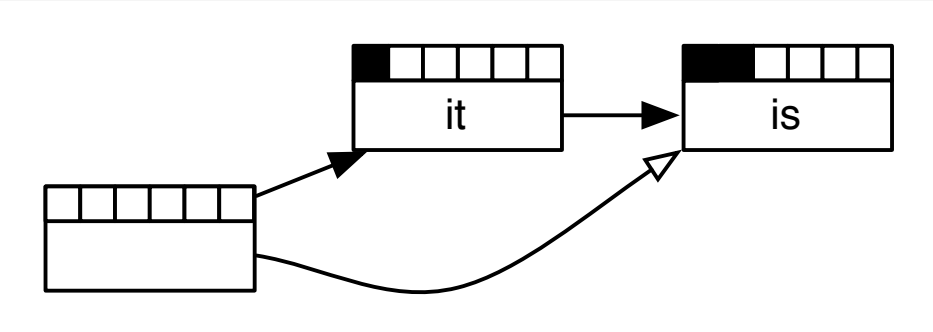


case 1:

- the same number of foreign words translated,
- the same English words in output:

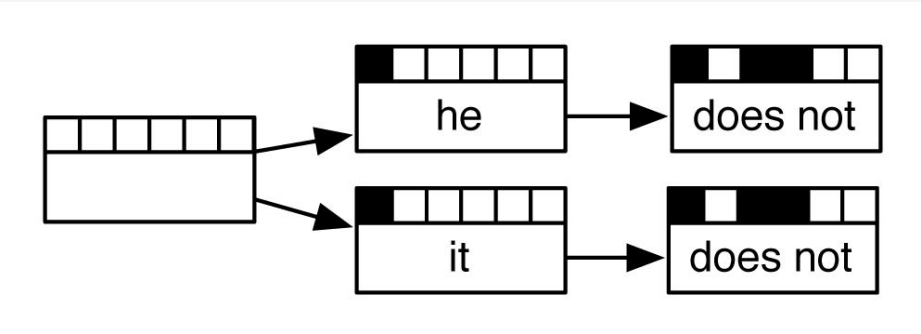


$\Rightarrow$ Drop the hypothesis with the worse score:



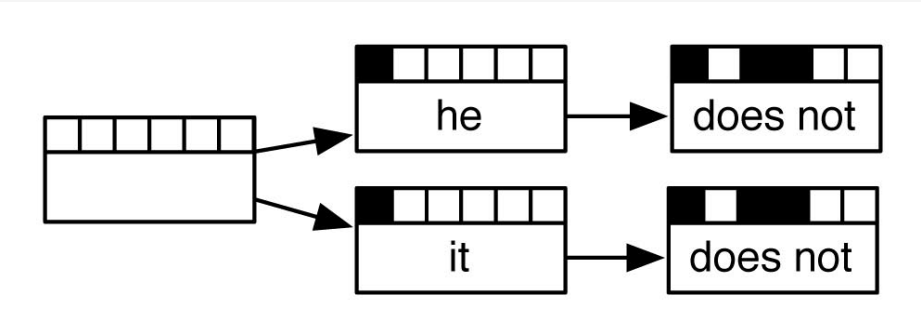
case 2:

- the same number of foreign words translated,
- the same last two words in output (assuming trigram lm),
- the same last foreign word translated:

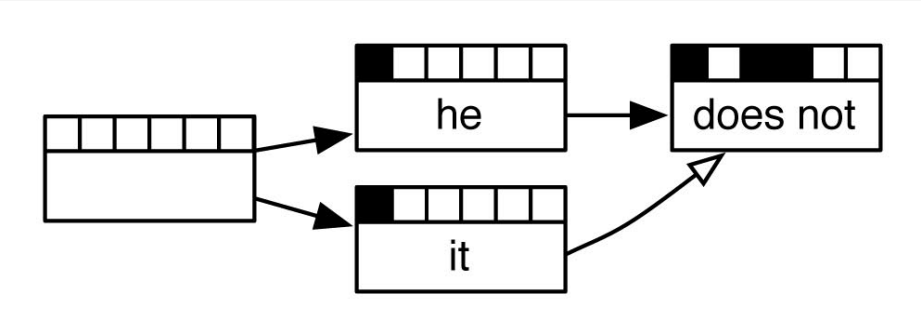


case 2:

- the same number of foreign words translated,
- the same last two words in output (assuming trigram lm),
- the same last foreign word translated:

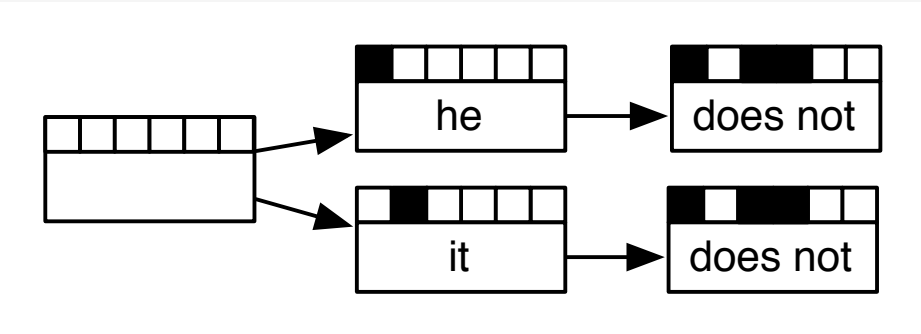


⇒ Drop the hypothesis with the worse score:



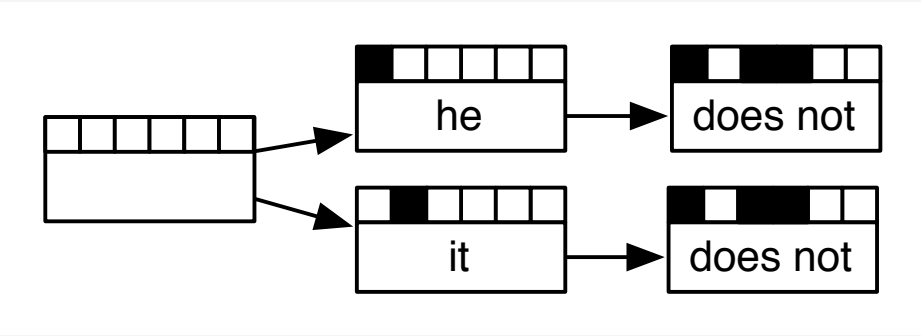
case 2:

- the same number of foreign words translated,
- the same last two words in output (assuming trigram lm),
- the same last foreign word translated:

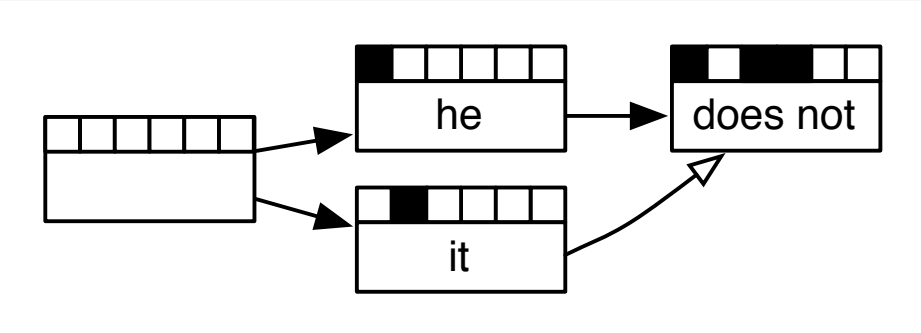


case 2:

- the same number of foreign words translated,
- the same last two words in output (assuming trigram lm),
- the same last foreign word translated:



⇒ Drop the hypothesis with the worse score:



- **translation model:** phrase translation independent from each other

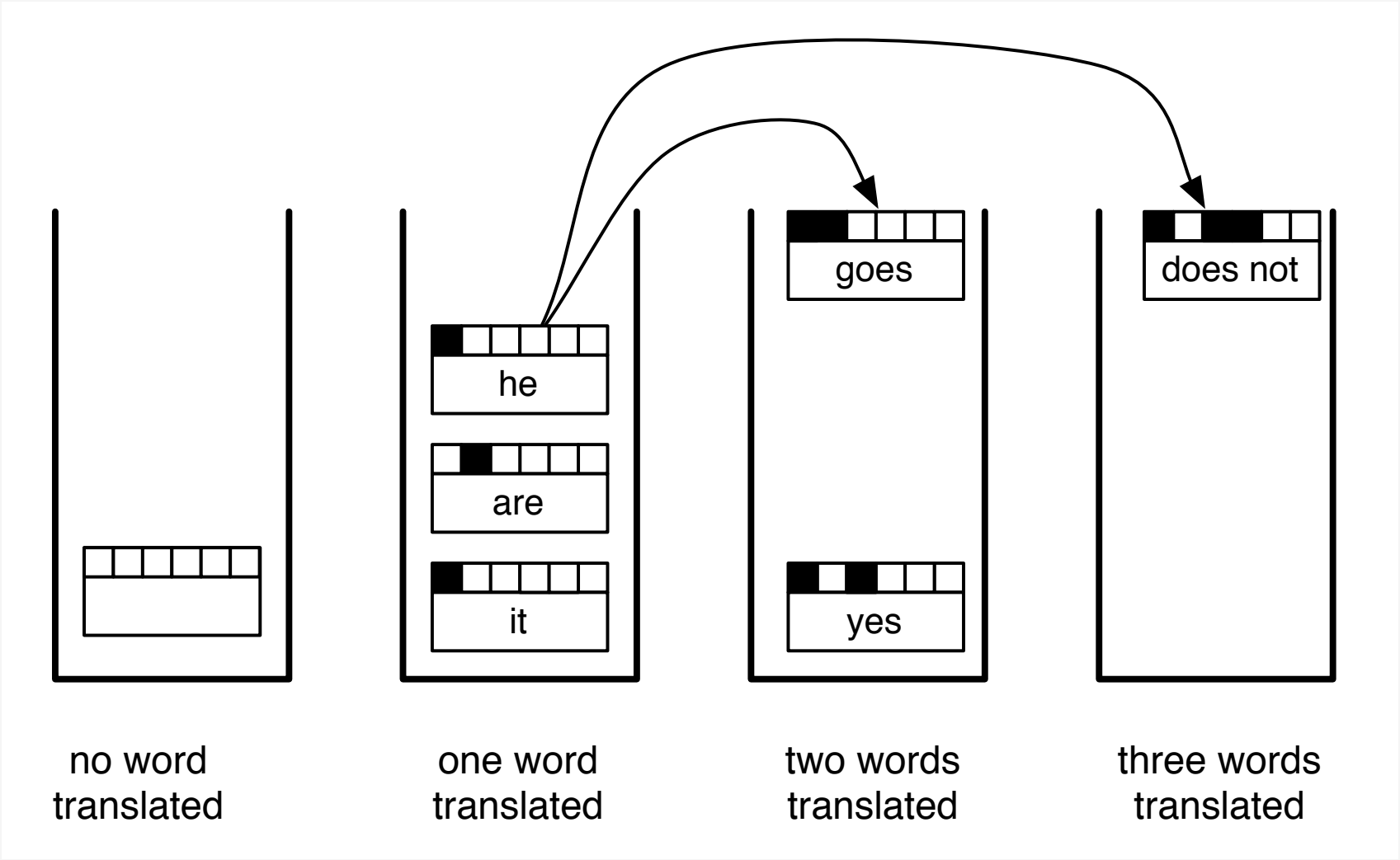
  $\Rightarrow$ no restriction to hypothesis recombination
- **language model:** last $n-1$ words used as history in n-gram LM

  $\Rightarrow$ 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
- other feature function may introduce additional restrictions

- recombination reduces search space, but not enough

  (still an NP complete problem)
- 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



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

```
1: place empty hypothesis into stack 0
2: for all stacks 0...n − 1 do
3:    for all hypotheses in stack do
4:       for all translation options do
5:          if applicable then
6:             create new hypothesis
7:             place in stack
8:             recombine with existing hypothesis if possible
9:             prune stack if too big
10:         end if
11:      end for
12:   end for
13: end for
```

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

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

- —translation options— is linear with sentence length, so

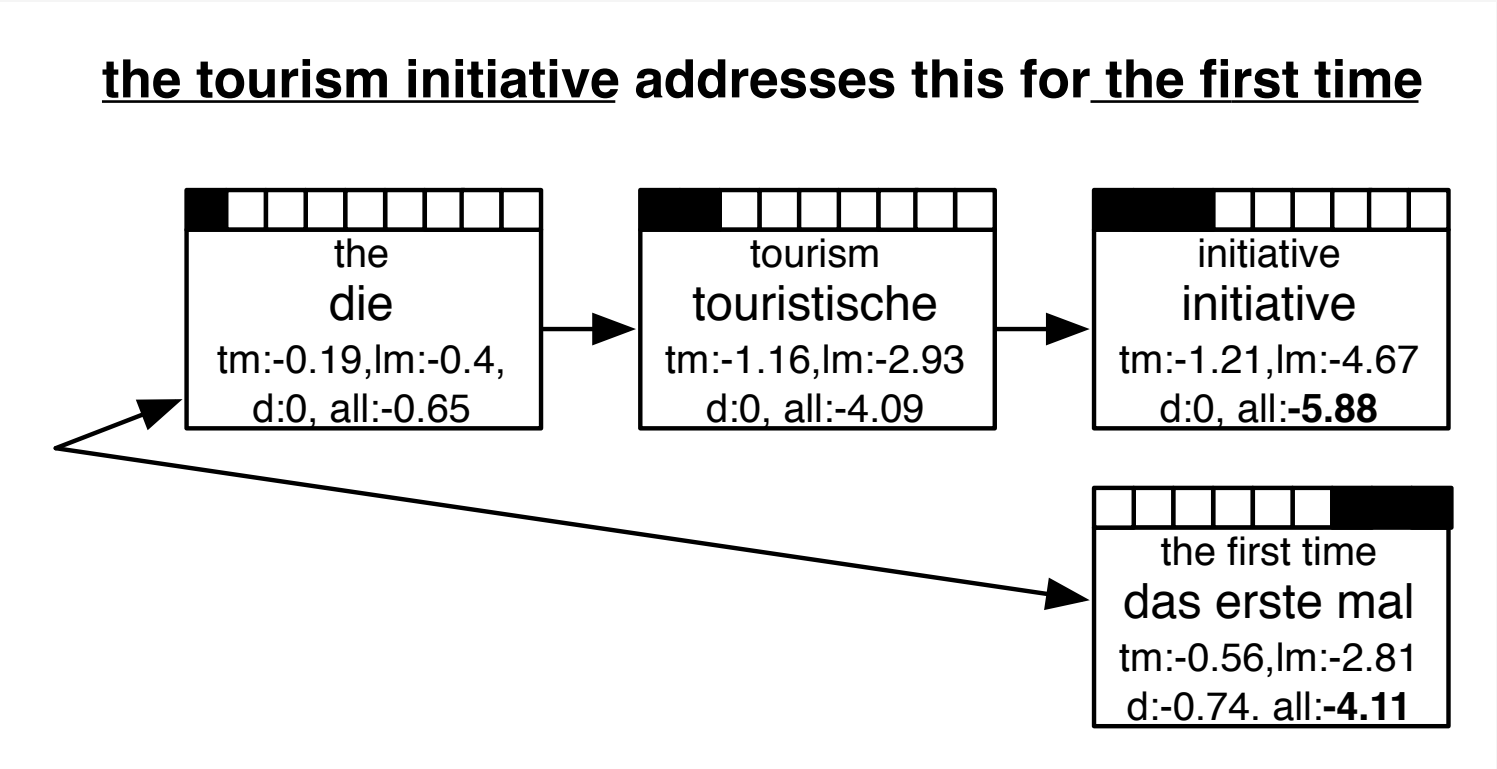
$$O(\text{max stack size} \times \text{sentence length}^2)$$

- polynomial!

- 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 (hiding $\simeq$constant # of translations options into O-symbol)

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

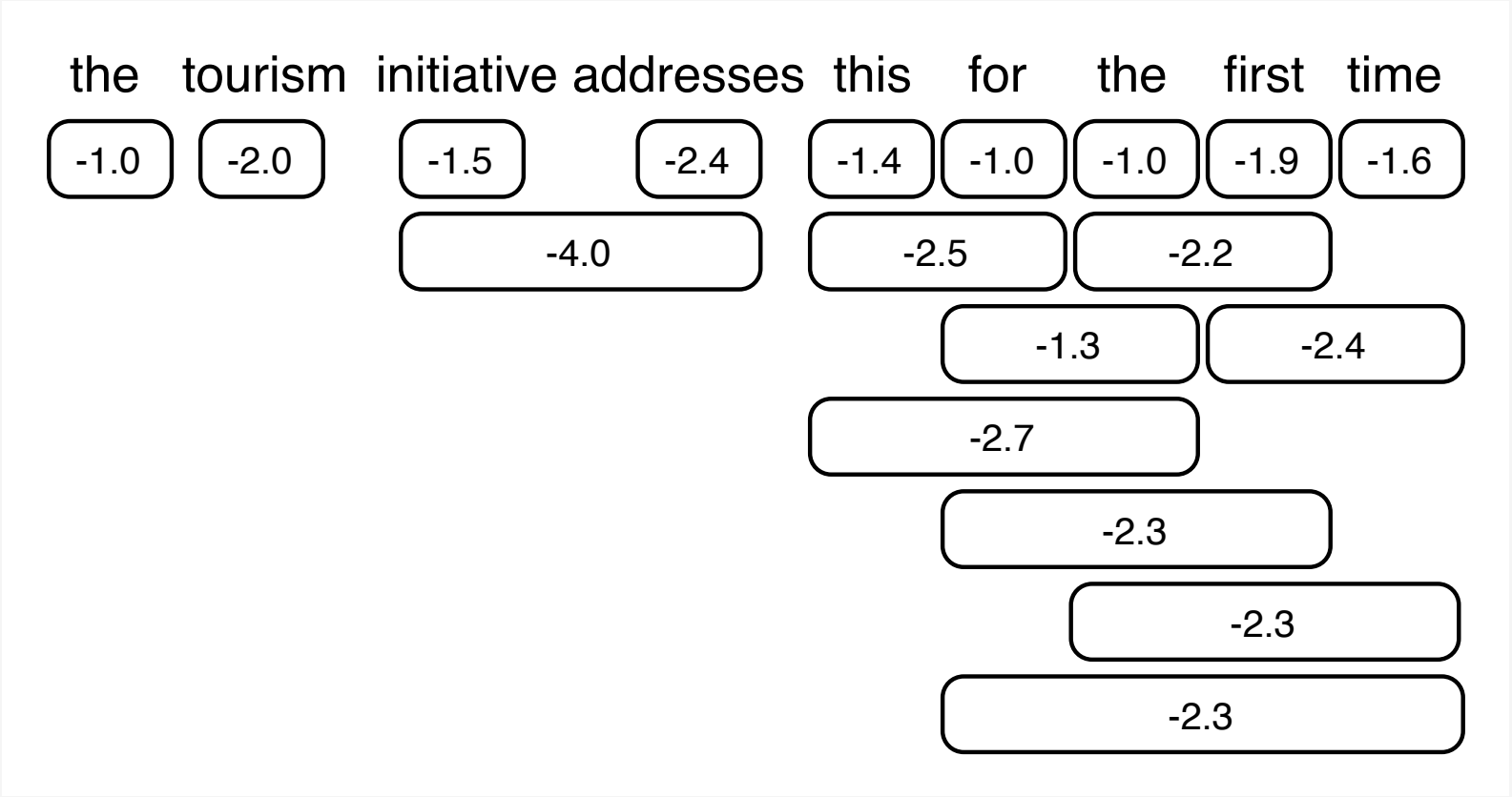
- speed / quality trade-off by setting maximum stack size



both hypotheses translate 3 words
worse hypothesis has better score

- 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

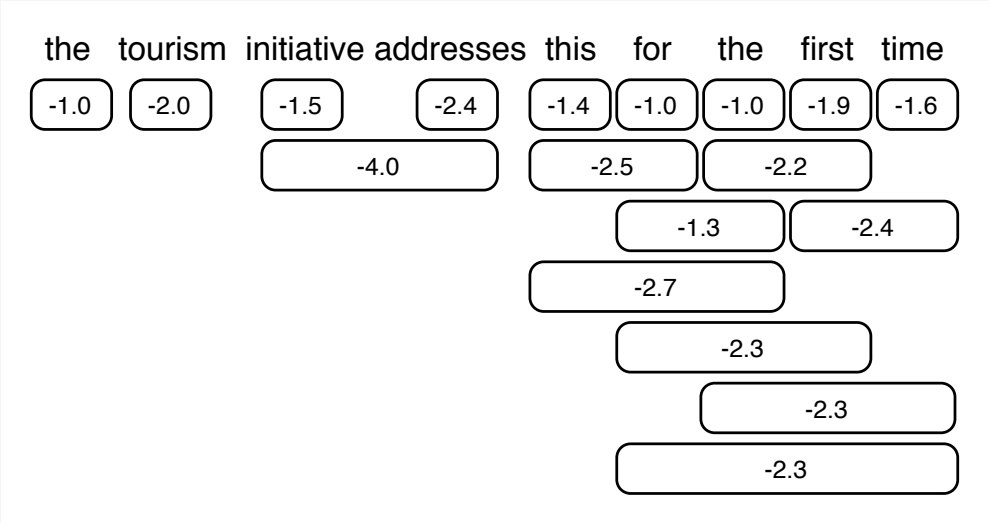
Cost Estimates from Translation Options



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

- function words cheaper (the: -1.0) than content words (tourism -2.0)
- common phrases cheaper (for the first time: -2.3)
  than unusual ones (initiative addresses: -4.0)

fill the table with initial probabilites

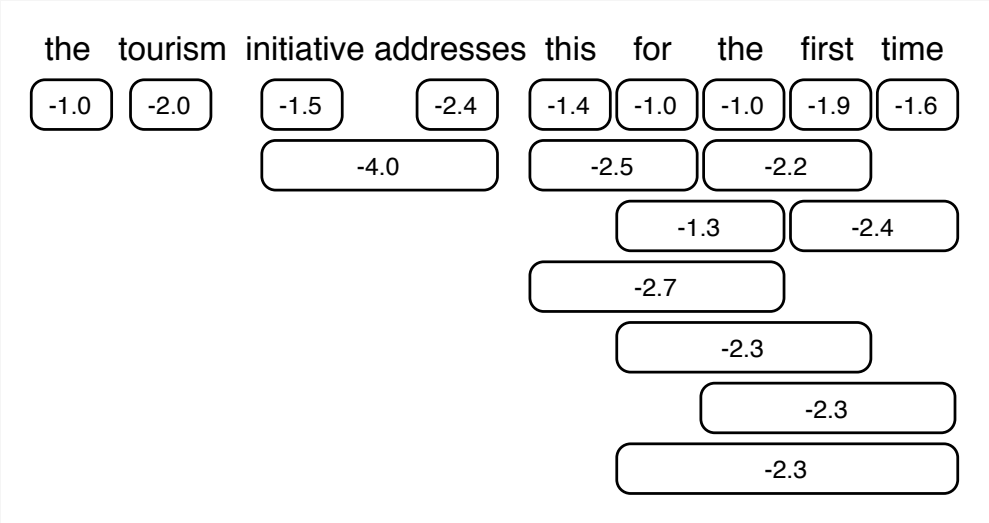


first word	future cost estimate for n words (from first)								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
the	-1.0								
tourism	-2.0								
initiative	-1.5	-4.0							
addresses	-2.4								
this	-1.4	-2.5	-2.7						
for	-1.0	-1.3	-2.3	-2.3					
the	-1.0	-2.2	-2.3						
first	-1.9	-2.4							
time	-1.6								

```
 1: for length = 1 .. n do
 2:   for start = 1...n+1-length do
 3:     end = start+length
 4:     cost(start,end) = infinity
 5:     cost(start,end) = translation option cost estimate if exists
 6:     for i=start..end-1 do
 7:       if cost(start,i) + cost(i+1,end) < cost(start,end) then
 8:         update cost(start,end)
 9:       end if
10:     end for
11:   end for
12: end for
```

Cost Estimates for all Spans

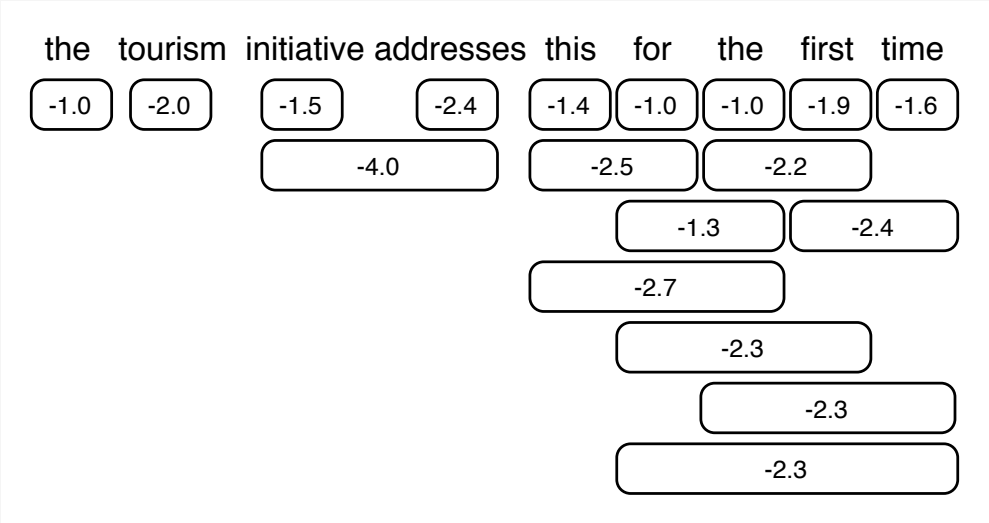
cost estimate for all contiguous spans by combining cheapest options



first word	future cost estimate for n words (from first)								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
the	-1.0								
tourism	-2.0								
initiative	-1.5	-4.0							
addresses	-2.4								
this	-1.4	-2.5	-2.7						
for	-1.0	-1.3	-2.3	-2.3					
the	-1.0	-2.2	-2.3						
first	-1.9	-2.4							
time	-1.6								

Cost Estimates for all Spans

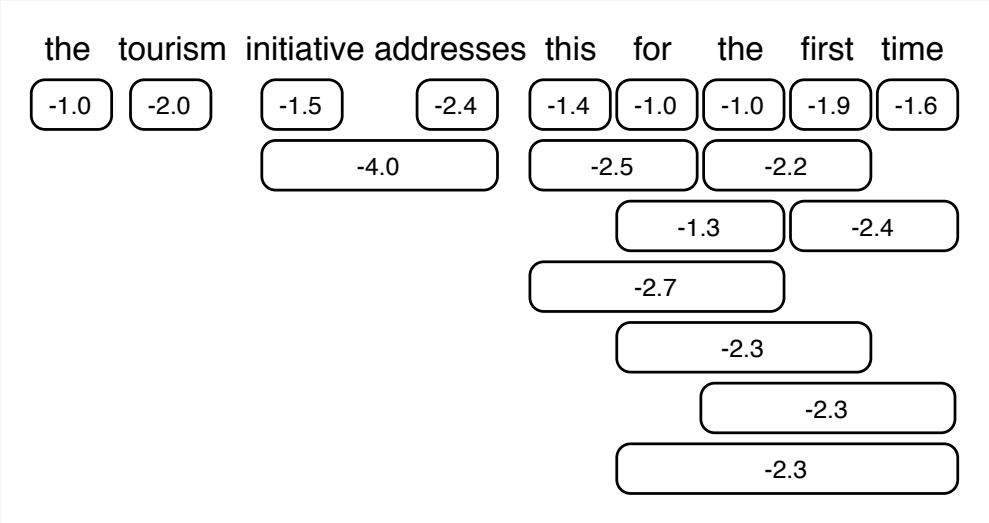
cost estimate for all contiguous spans by combining cheapest options



first word	future cost estimate for n words (from first)								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
the	-1.0	-3.0							
tourism	-2.0								
initiative	-1.5	-4.0							
addresses	-2.4								
this	-1.4	-2.5	-2.7						
for	-1.0	-1.3	-2.3	-2.3					
the	-1.0	-2.2	-2.3						
first	-1.9	-2.4							
time	-1.6								

Cost Estimates for all Spans

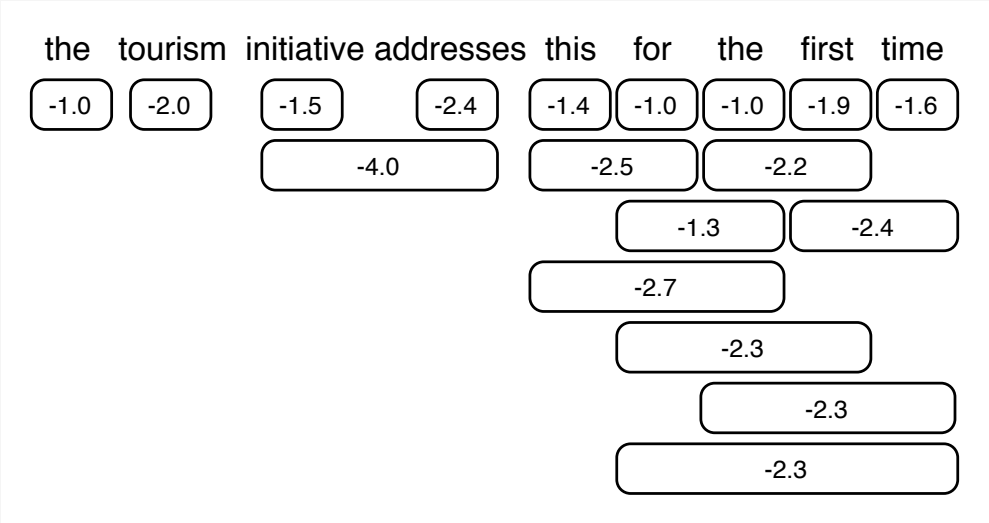
cost estimate for all contiguous spans by combining cheapest options



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						
tourism	-2.0								
initiative	-1.5	-4.0							
addresses	-2.4								
this	-1.4	-2.5	-2.7						
for	-1.0	-1.3	-2.3	-2.3					
the	-1.0	-2.2	-2.3						
first	-1.9	-2.4							
time	-1.6								

Cost Estimates for all Spans

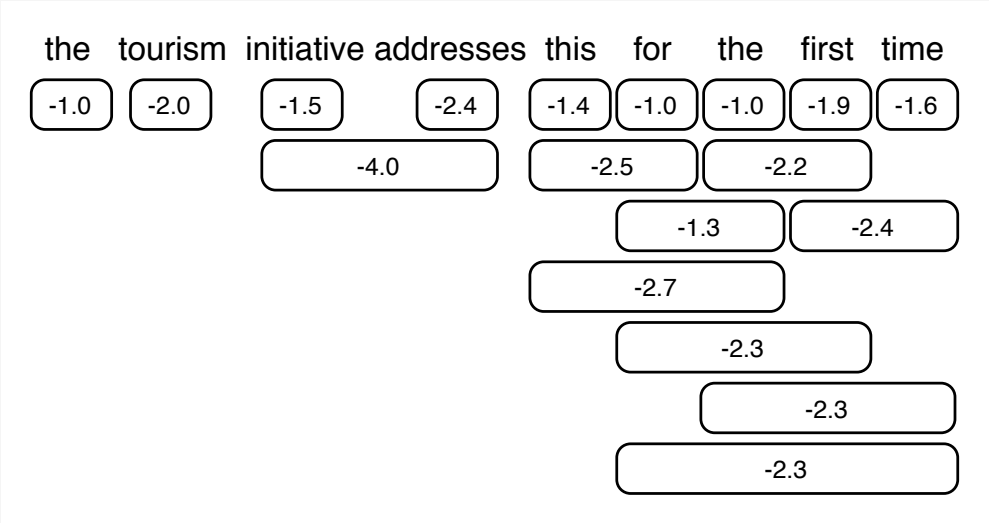
cost estimate for all contiguous spans by combining cheapest options



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						
tourism	-2.0								
initiative	-1.5	-3.9							
addresses	-2.4								
this	-1.4	-2.5	-2.7						
for	-1.0	-1.3	-2.3	-2.3					
the	-1.0	-2.2	-2.3						
first	-1.9	-2.4							
time	-1.6								

Cost Estimates for all Spans

cost estimate for all contiguous spans by combining cheapest options



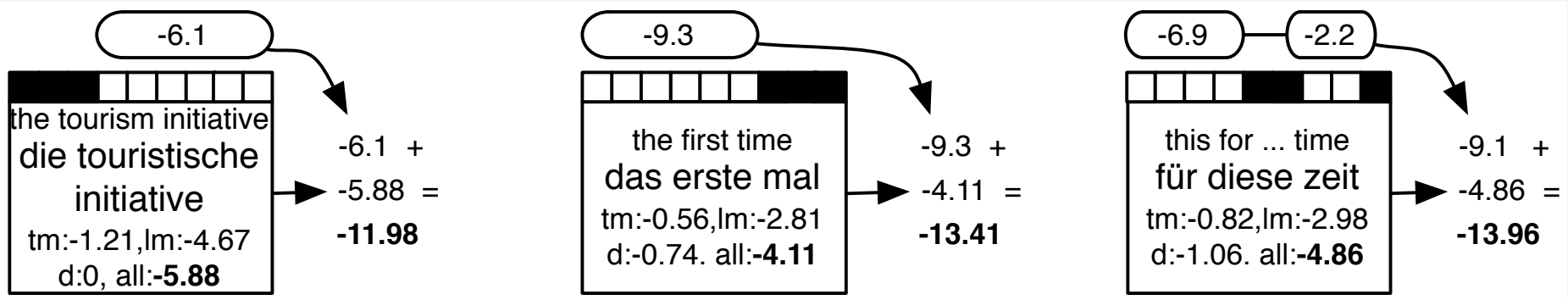
first word	future cost estimate for n words (from first)								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
the	-1.0	-3.0	-4.5						
tourism	-2.0								
initiative	-1.5	-3.9							
addresses	-2.4								
this	-1.4	-2.4	-2.7						
for	-1.0	-1.3	-2.3	-2.3					
the	-1.0	-2.2	-2.3						
first	-1.9	-2.4							
time	-1.6								

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