Syntax-Based Decoding

Philipp Koehn

9 November 2017

syntax-based models

Synchronous Context Free Grammar Rules

- Nonterminal rules

$$\mathsf{NP} \rightarrow \mathsf{DET}_1\ \mathsf{NN}_2\ \mathsf{JJ}_3 \mid \mathsf{DET}_1\ \mathsf{JJ}_3\ \mathsf{NN}_2$$

- Terminal rules

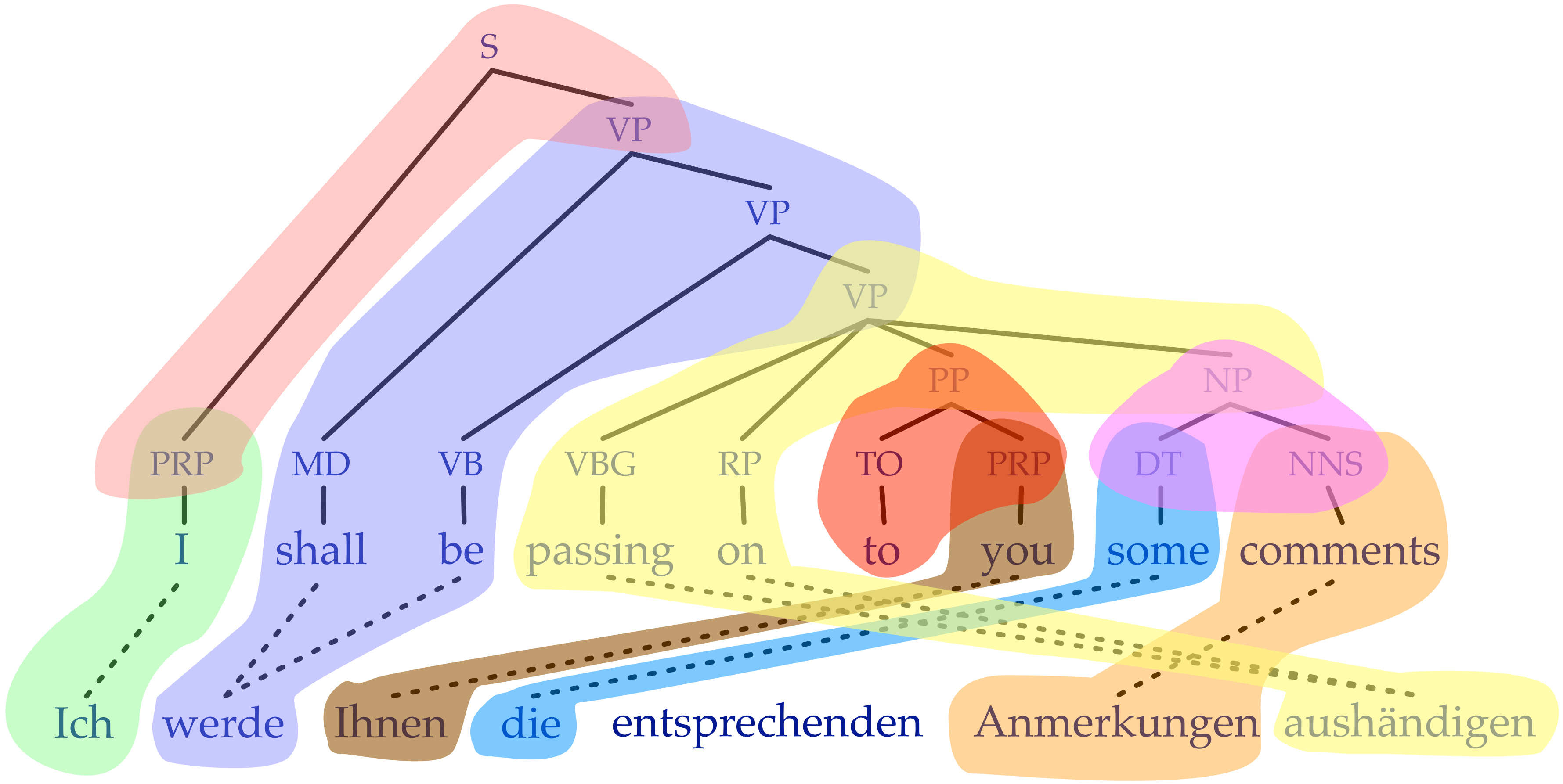
$$\mathsf{N} \rightarrow \text{maison} \mid \text{house}$$

$$\mathsf{NP} \rightarrow \text{la maison bleue} \mid \text{the blue house}$$

- Mixed rules

$$\mathsf{NP} \rightarrow \text{la maison } \mathsf{JJ}_1 \mid \text{the } \mathsf{JJ}_1 \text{ house}$$

Extracting Minimal Rules



Extracted rule: $\mathsf{S} \rightarrow \mathsf{X}_1\ \mathsf{X}_2 \mid \mathsf{PRP}_1\ \mathsf{VP}_2$

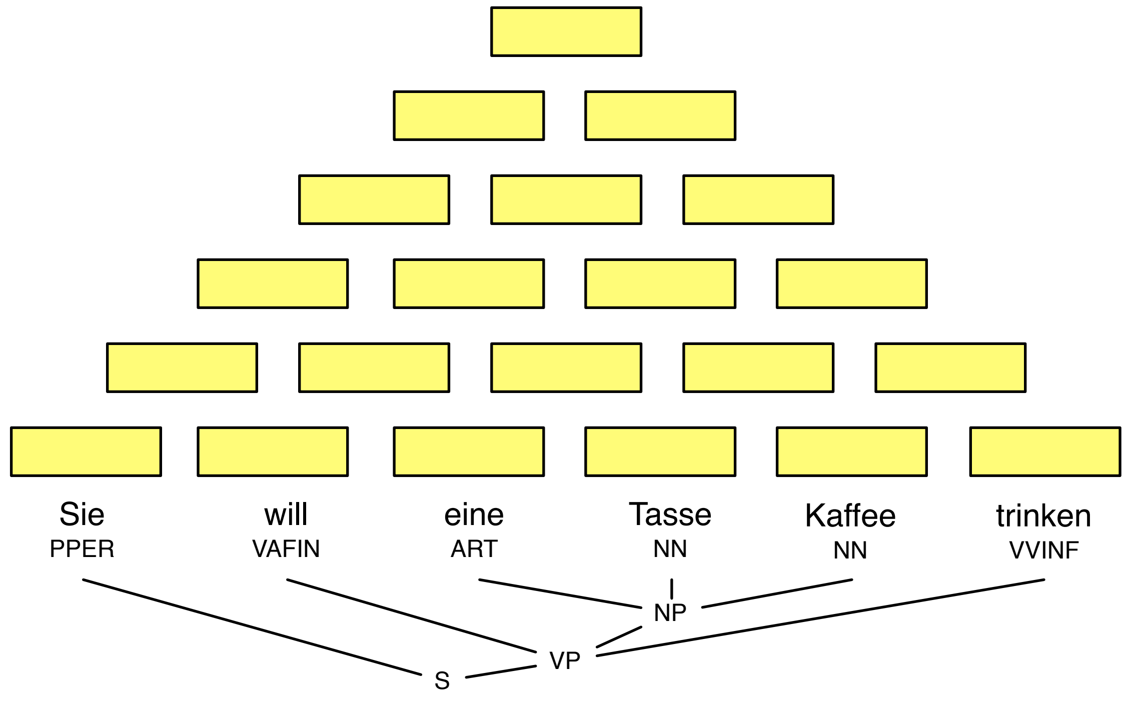
DONE — note: one rule per alignable constituent

decoding

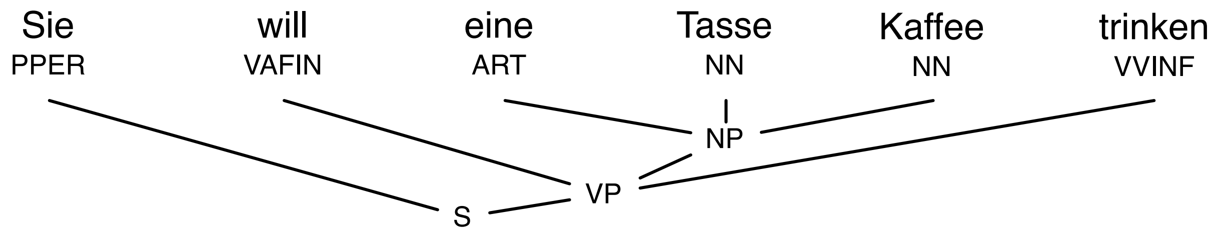
Syntactic Decoding

Inspired by monolingual syntactic chart parsing:

During decoding of the source sentence,
a chart with translations for the $O(n^2)$ spans has to be filled

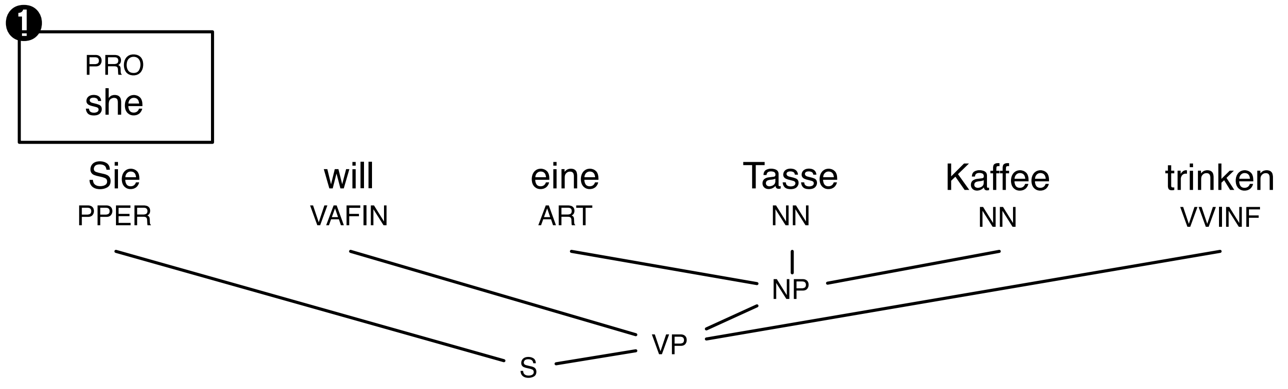


Syntax Decoding



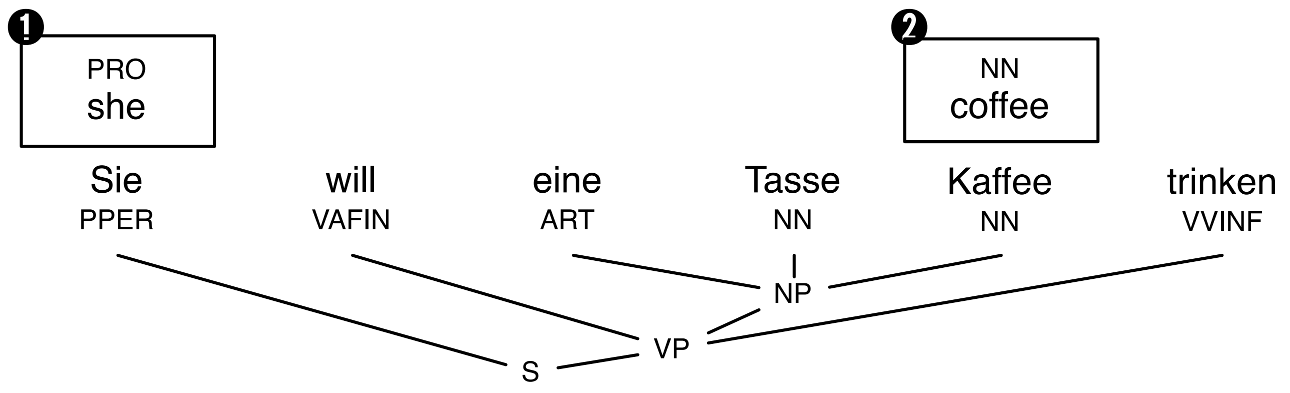
German input sentence with tree

Syntax Decoding



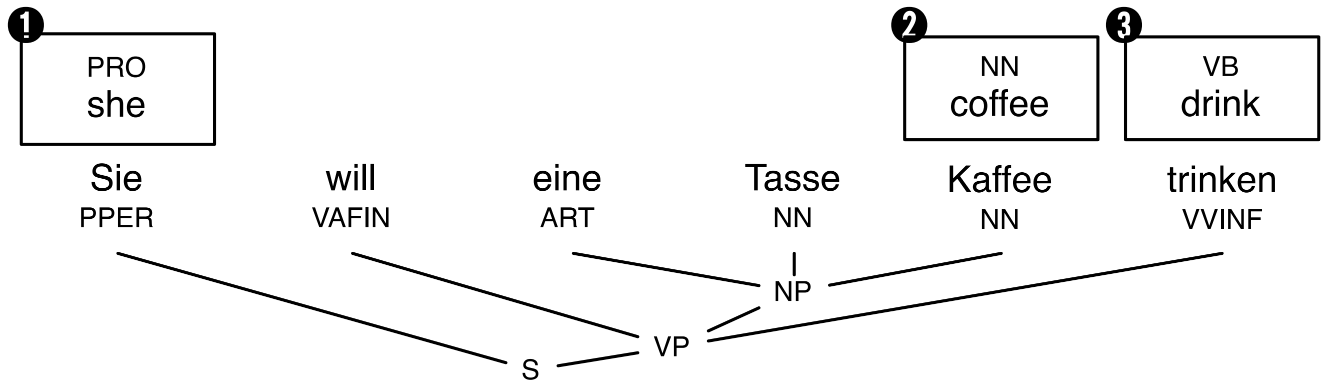
Purely lexical rule: filling a span with a translation (a constituent in the chart)

Syntax Decoding



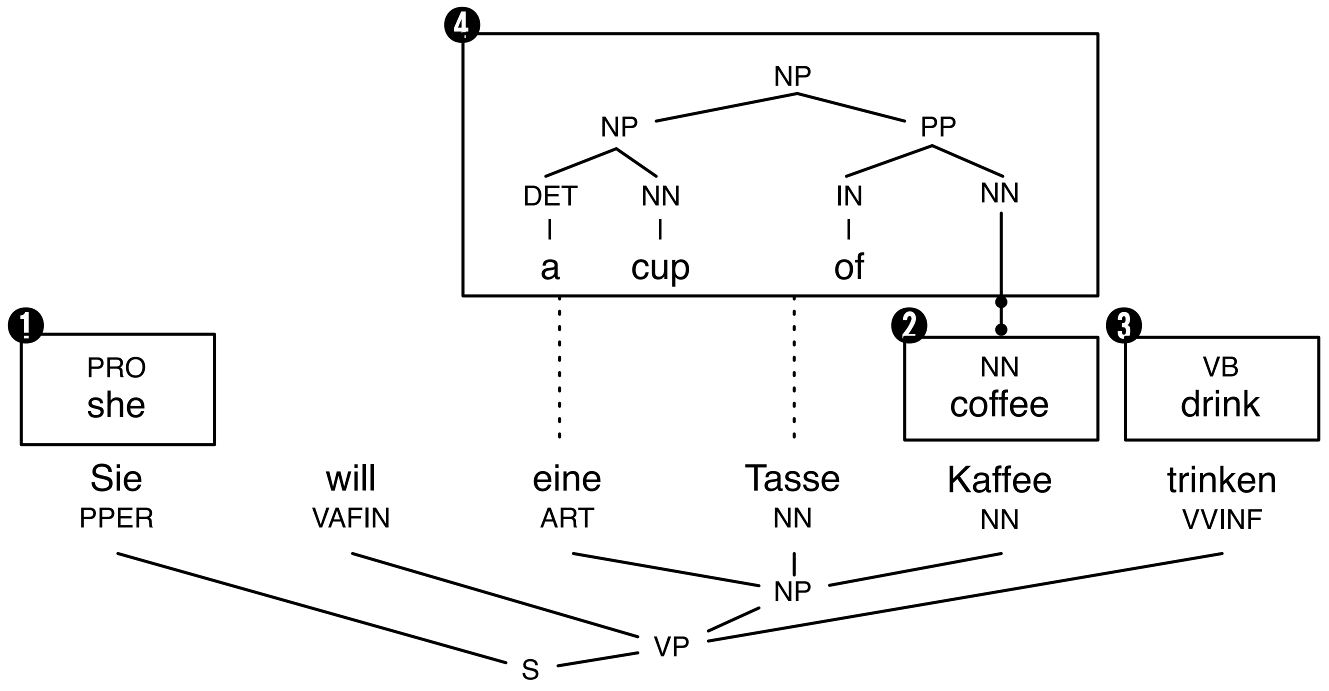
Purely lexical rule: filling a span with a translation (a constituent in the chart)

Syntax Decoding



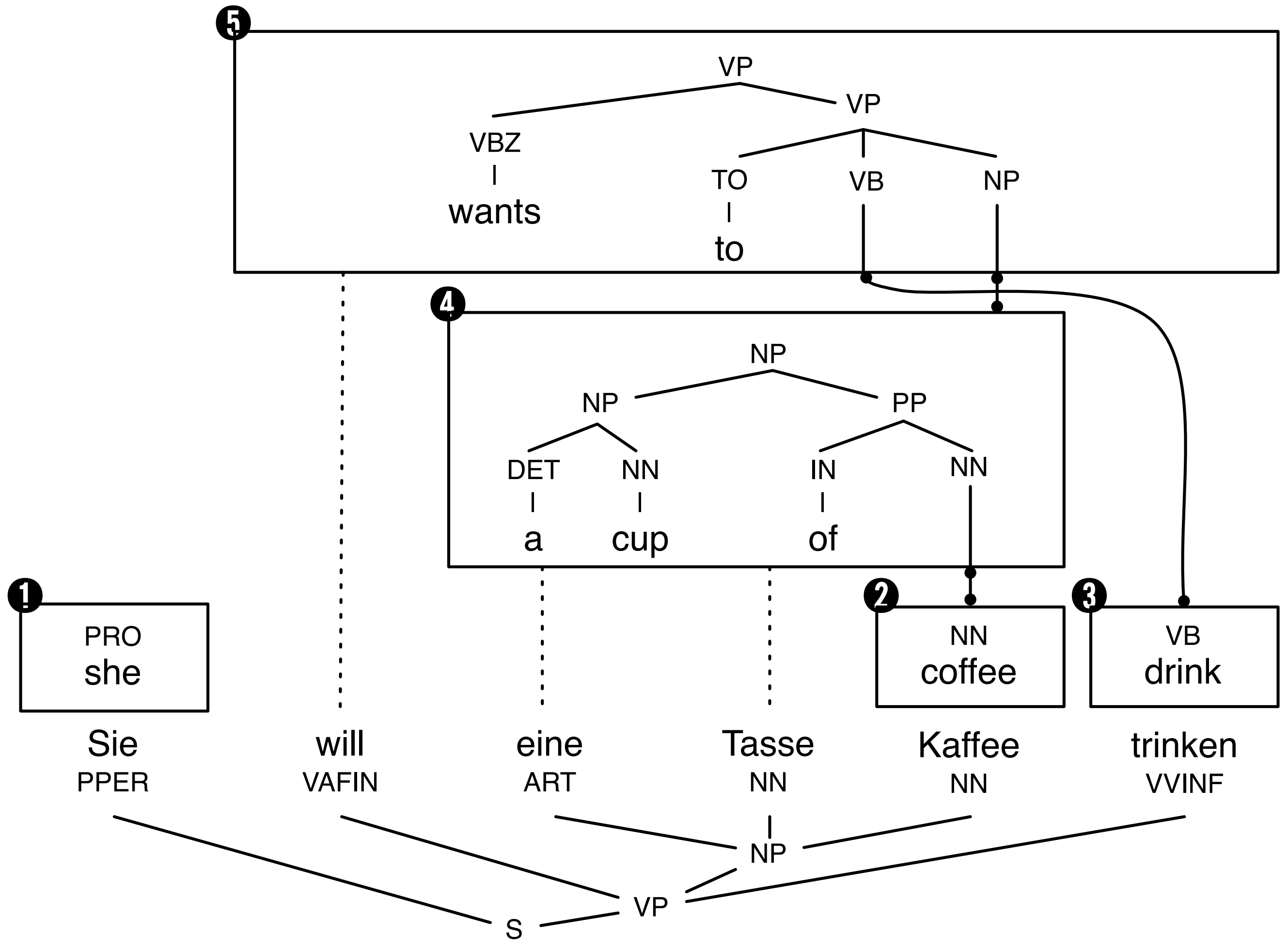
Purely lexical rule: filling a span with a translation (a constituent in the chart)

Syntax Decoding



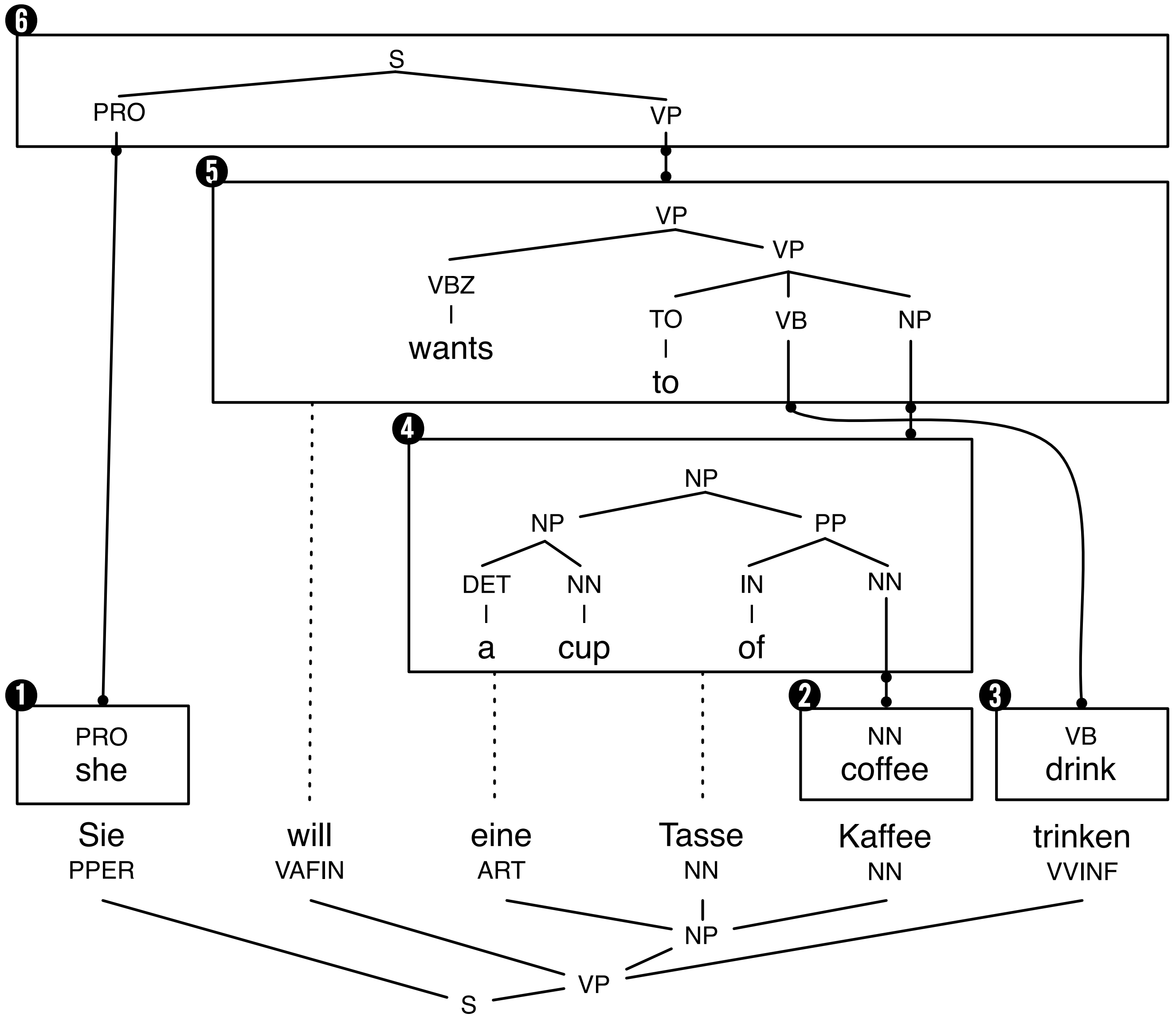
Complex rule: matching underlying constituent spans, and covering words

Syntax Decoding



Complex rule with reordering

Syntax Decoding



Bottom-Up Decoding

- For each span, a stack of (partial) translations is maintained
- Bottom-up: a higher stack is filled, once underlying stacks are complete

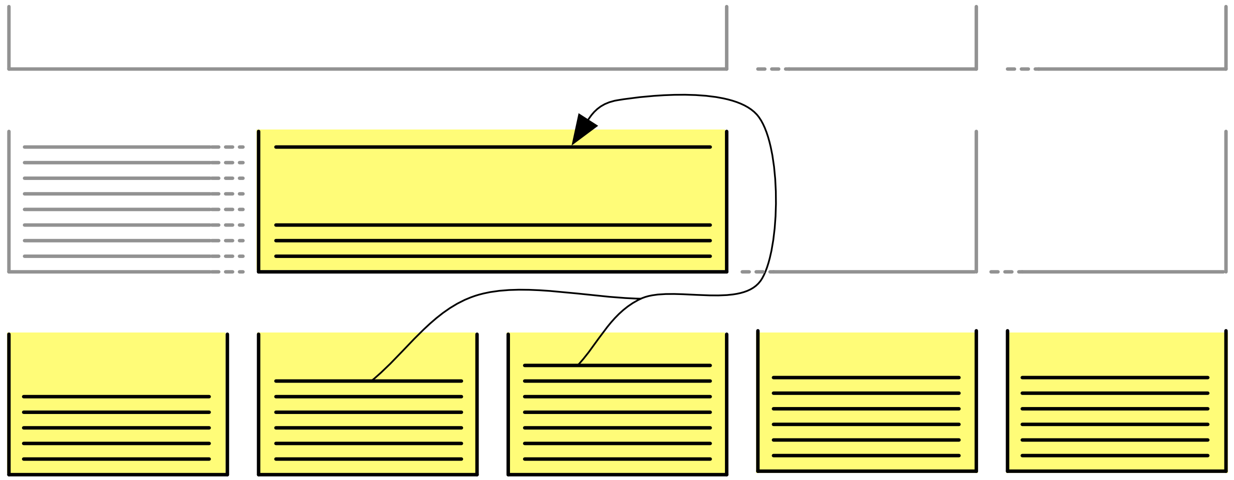
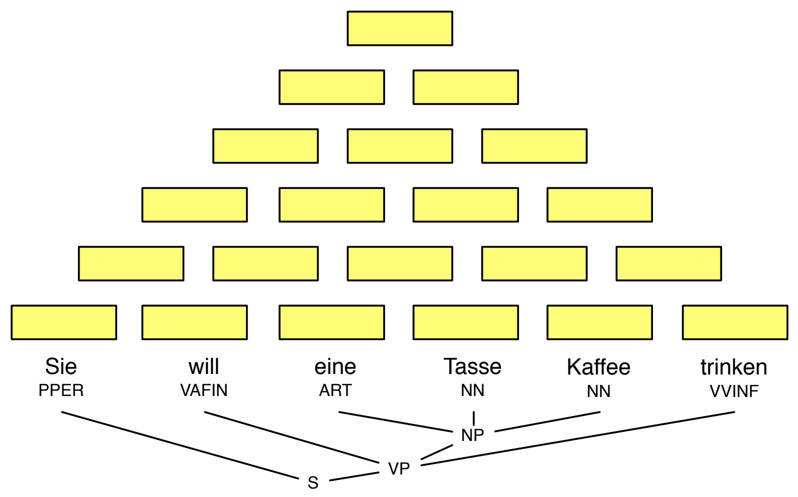
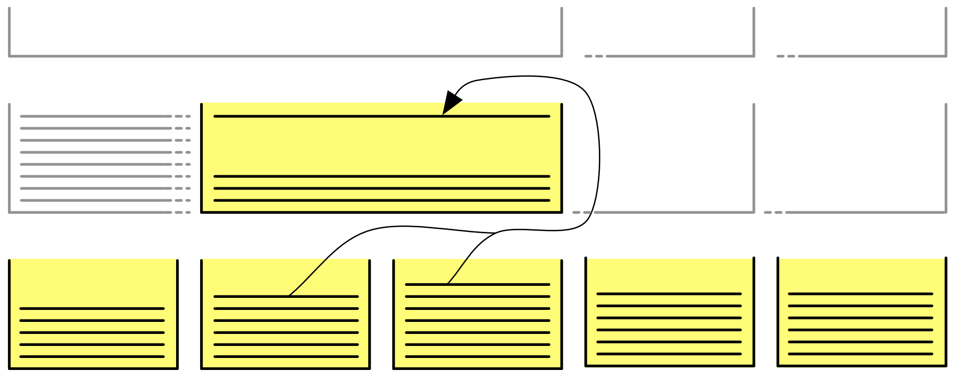


Chart Organization





- Chart consists of cells that cover contiguous spans over the input sentence
- Each cell contains a set of hypotheses[1]
- Hypothesis = translation of span with target-side constituent

[1] In the book, they are called chart entries.

Naive Algorithm

Input: Foreign sentence $\mathbf{f} = f_1, ...f_{l_f}$, with syntax tree
Output: English translation $\mathbf{e}$

```
 1: for all spans [start,end] (bottom up) do
 2:     for all sequences $s$ of hypotheses and words in span [start,end] do
 3:         for all rules $r$ do
 4:             if rule $r$ applies to chart sequence $s$ then
 5:                 create new hypothesis $c$
 6:                 add hypothesis $c$ to chart
 7:             end if
 8:         end for
 9:     end for
10: end for
11: return English translation $\mathbf{e}$ from best hypothesis in span [0,$l_f$]
```

Stack Pruning

- Number of hypotheses in each chart cell explodes

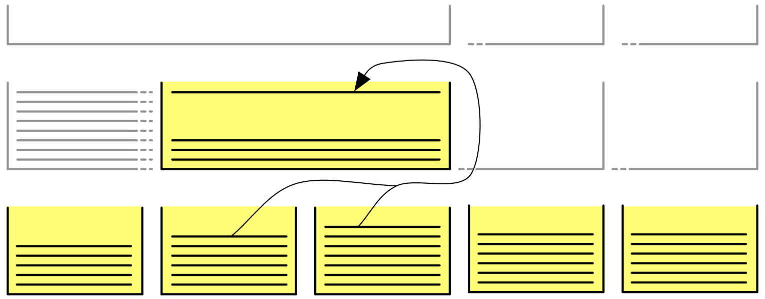
- Dynamic programming (recombination) not enough

⇒ need to discard bad hypotheses
e.g., keep 100 best only

- Different stacks for different output constituent labels?

- Cost estimates
  - translation model cost known
  - language model cost for internal words known
    → estimates for initial words
  - outside cost estimate?
    (how useful will be a NP covering input words 3–5 later on?)

Naive Algorithm: Blow-ups



- Many subspan sequences

  for all sequences s of hypotheses and words in span [start,end]

- Many rules

  for all rules r

- Checking if a rule applies not trivial

  rule r applies to chart sequence s

$\Rightarrow$ Unworkable

Solution

- Prefix tree data structure for rules
- Dotted rules
- Cube pruning

storing rules efficiently

Storing Rules

- First concern: do they apply to span?
  $\rightarrow$ have to match available hypotheses and input words

- Example rule

$$\textsf{NP} \rightarrow \textsf{X}_1 \text{ des } \textsf{X}_2 \mid \textsf{NP}_1 \text{ of the } \textsf{NN}_2$$

- Check for applicability
  - is there an initial sub-span that with a hypothesis with constituent label NP?
  - is it followed by a sub-span over the word des?
  - is it followed by a final sub-span with a hypothesis with label NN?

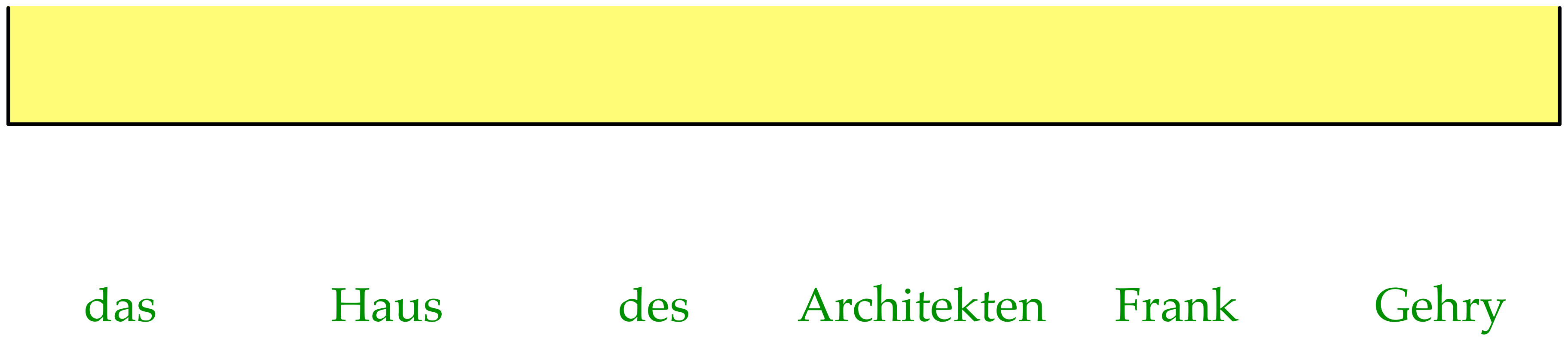
- Sequence of relevant information

$$\textsf{NP} \bullet \text{des} \bullet \textsf{NN} \bullet \textsf{NP}_1 \text{ of the } \textsf{NN}_2$$

Rule Applicability Check

Trying to cover a span of six words with given rule

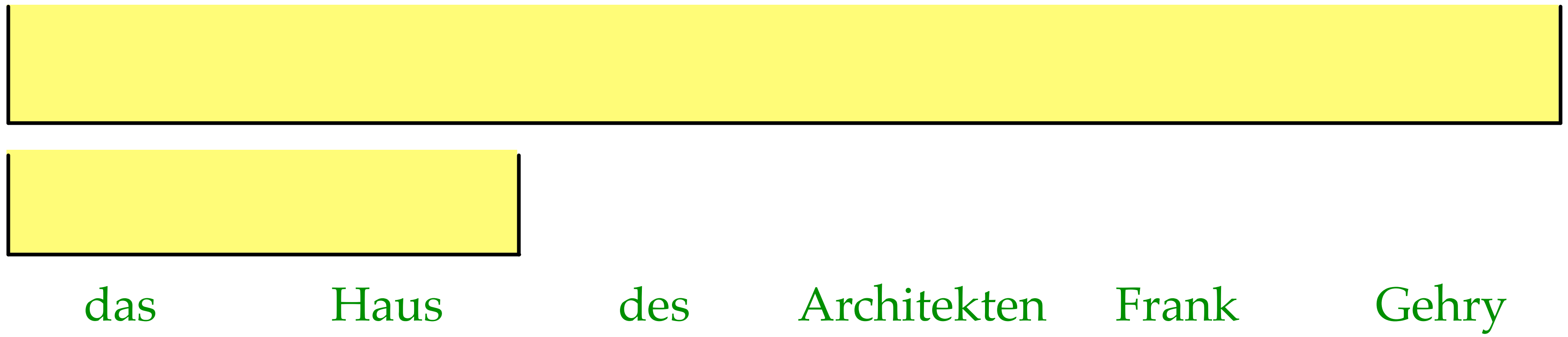
NP • des • NN → NP: NP of the NN



Rule Applicability Check

First: check for hypotheses with output constituent label NP

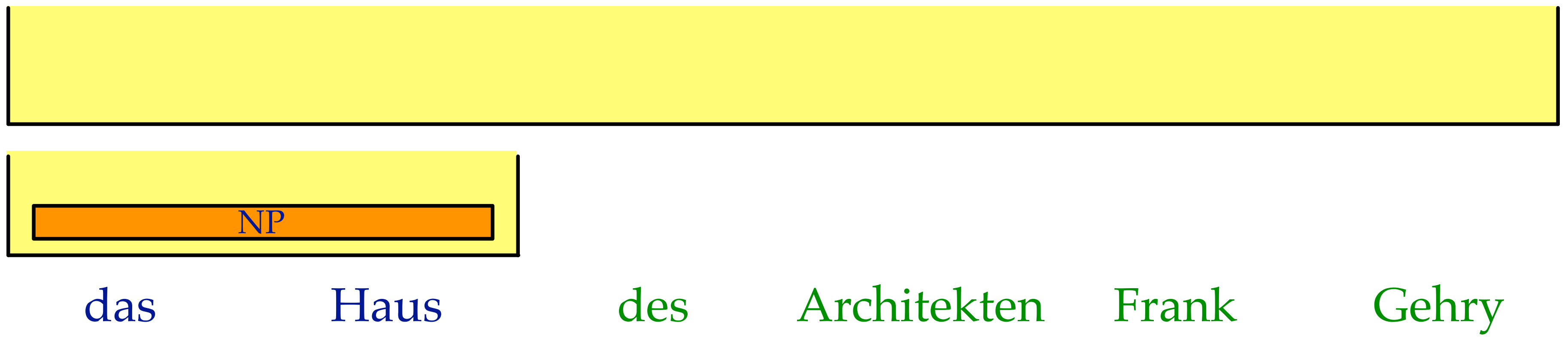
NP • des • NN → NP: NP of the NN



Rule Applicability Check

Found NP hypothesis in cell, matched first symbol of rule

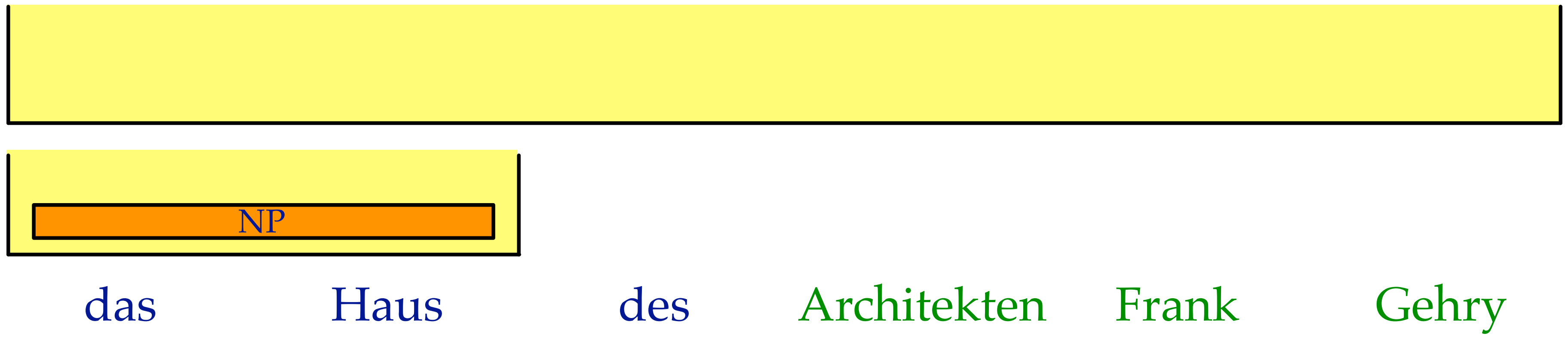
NP • des • NN → NP: NP of the NN



Rule Applicability Check

Matched word des, matched second symbol of rule

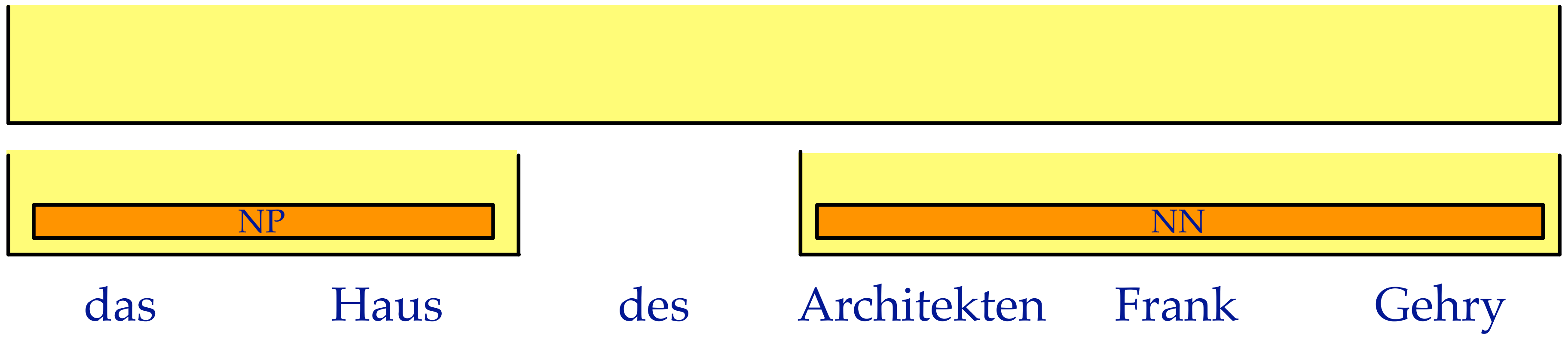
NP • des • NN → NP: NP of the NN



Rule Applicability Check

Found a NN hypothesis in cell, matched last symbol of rule

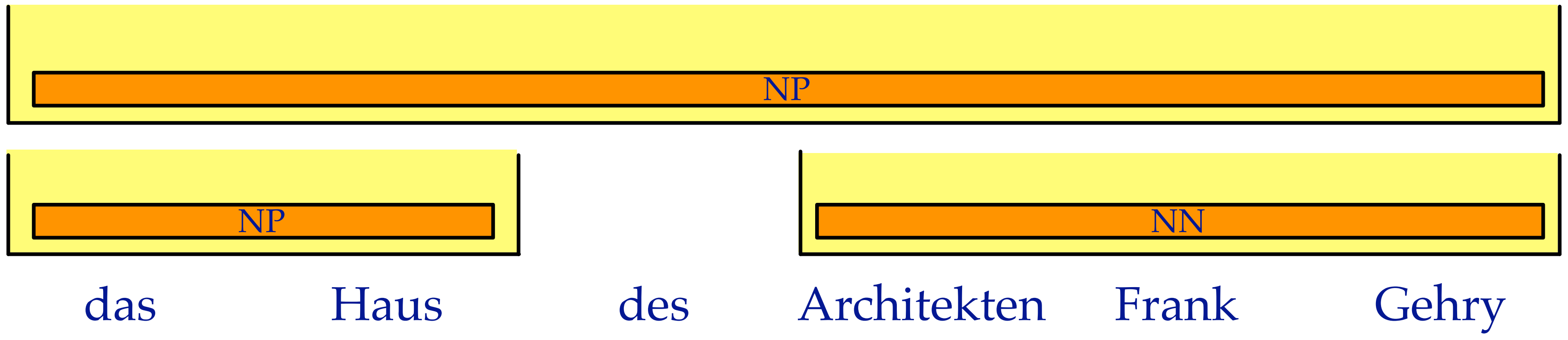
NP • des • NN → NP: NP of the NN



Rule Applicability Check

Matched entire rule → apply to create a NP hypothesis

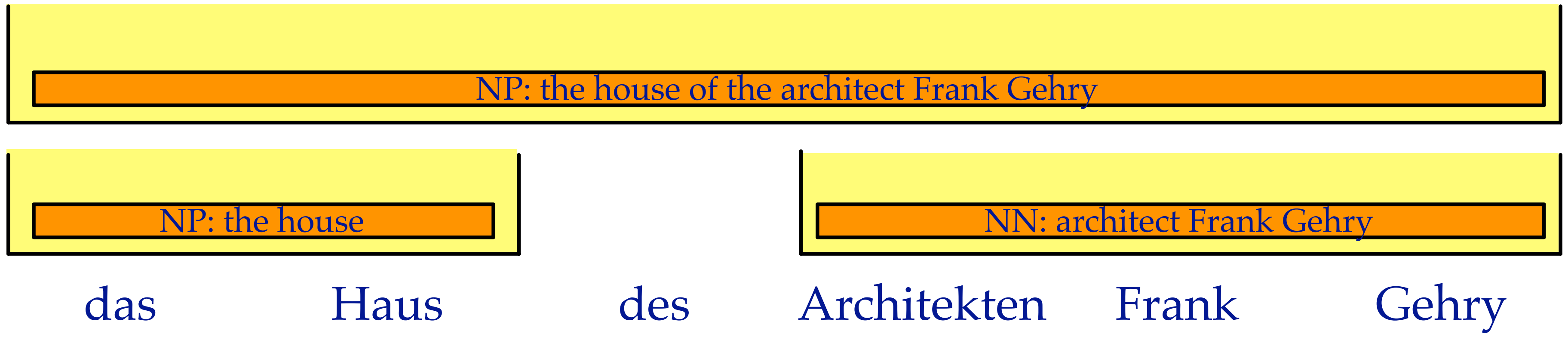
NP • des • NN → NP: NP of the NN



Rule Applicability Check

Look up output words to create new hypothesis
(note: there may be many matching underlying NP and NN hypotheses)

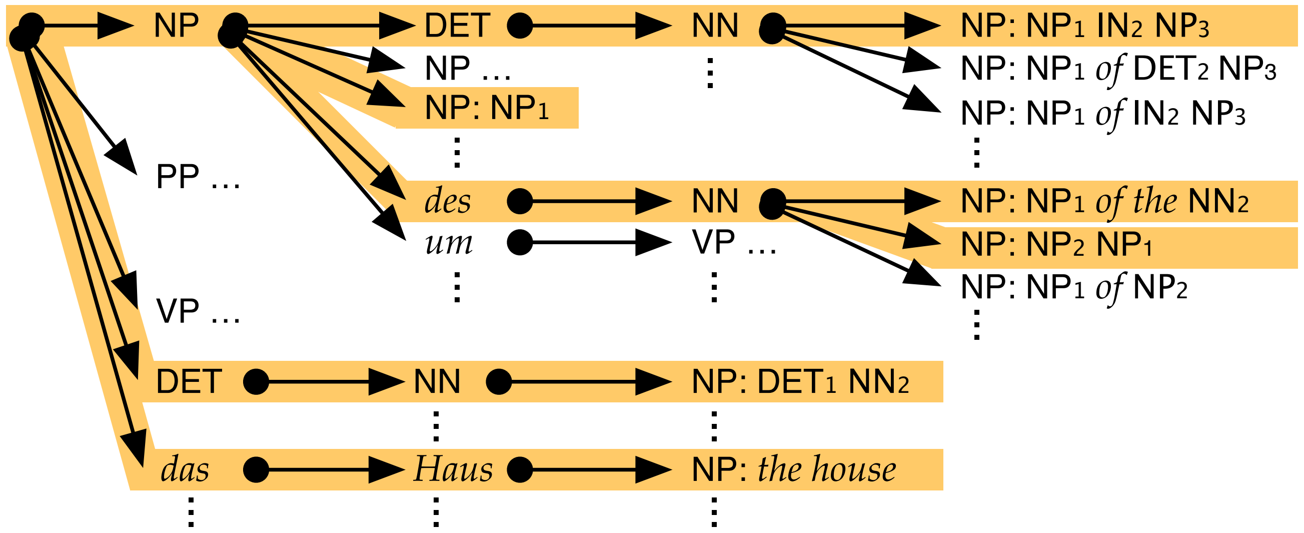
NP • des • NN → NP: NP of the NN


NP: the house of the architect Frank Gehry
NP: the house
NN: architect Frank Gehry
das
Haus
des
Architekten
Frank
Gehry

Checking Rules vs. Finding Rules

- What we showed:
  - given a rule
  - check if and how it can be applied
- But there are too many rules (millions) to check them all
- Instead:
  - given the underlying chart cells and input words
  - find which rules apply

Prefix Tree for Rules



Highlighted Rules

$\text{NP} \rightarrow \text{NP}_1\ \text{DET}_2\ \text{NN}_3 \mid \text{NP}_1\ \text{IN}_2\ \text{NN}_3$

$\text{NP} \rightarrow \text{NP}_1 \mid \text{NP}_1$

$\text{NP} \rightarrow \text{NP}_1\ \text{des}\ \text{NN}_2 \mid \text{NP}_1\ \text{of the}\ \text{NN}_2$

$\text{NP} \rightarrow \text{NP}_1\ \text{des}\ \text{NN}_2 \mid \text{NP}_2\ \text{NP}_1$

$\text{NP} \rightarrow \text{DET}_1\ \text{NN}_2 \mid \text{DET}_1\ \text{NN}_2$

$\text{NP} \rightarrow \text{das Haus} \mid \text{the house}$

dotted rules

Dotted Rules: Key Insight

- If we can apply a rule like

$$p \rightarrow A\ B\ C \mid x$$

  to a span

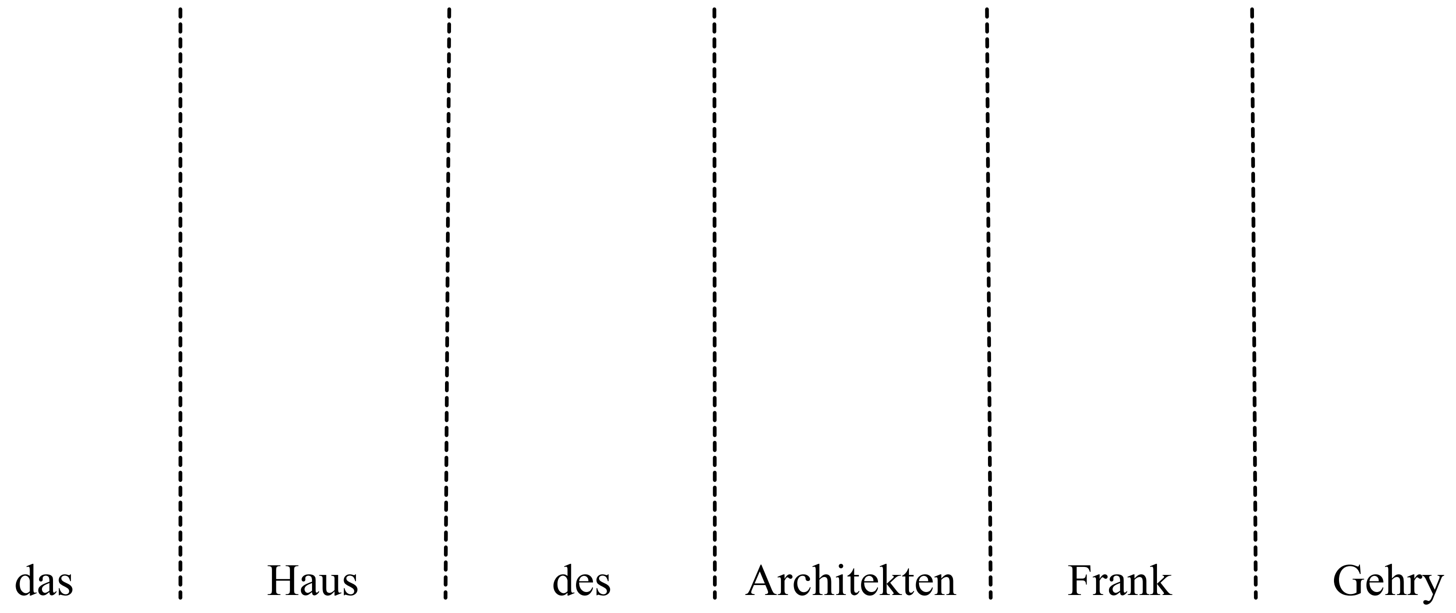
- Then we could have applied a rule like

$$q \rightarrow A\ B \mid y$$

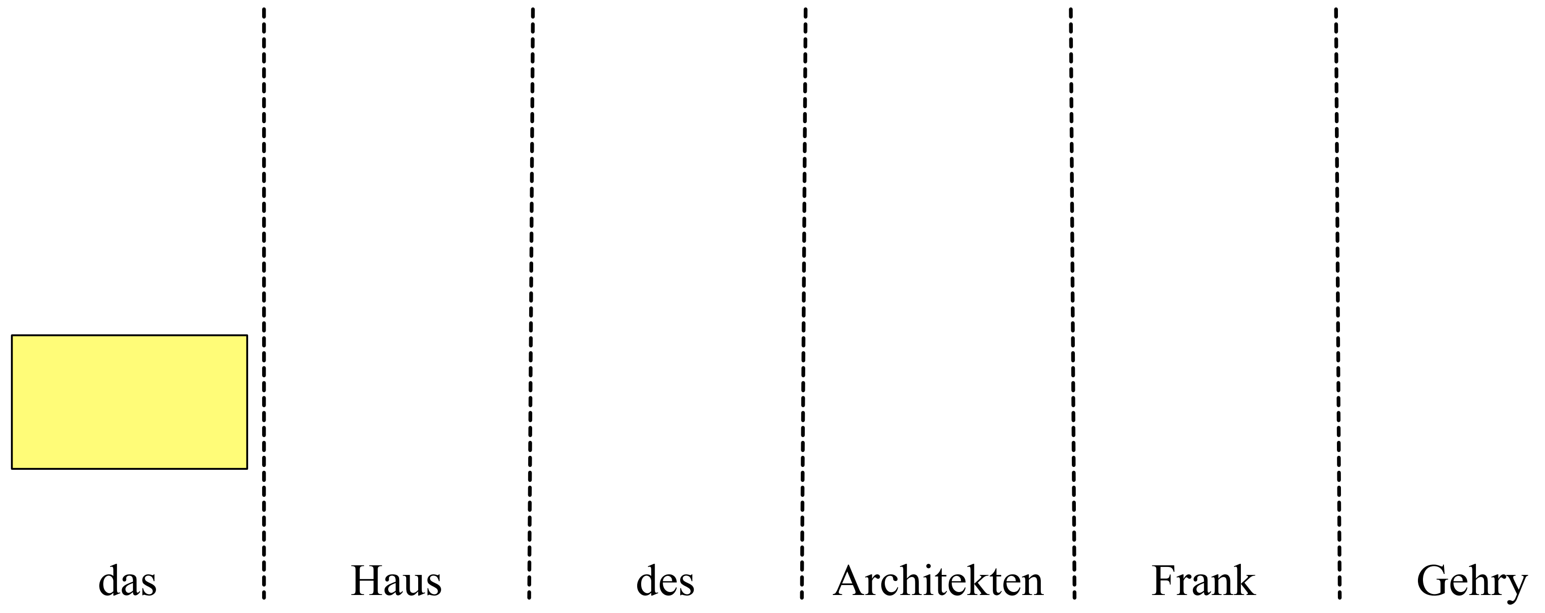
  to a sub-span with the same starting word

⇒ We can re-use rule lookup by storing A B • (dotted rule)

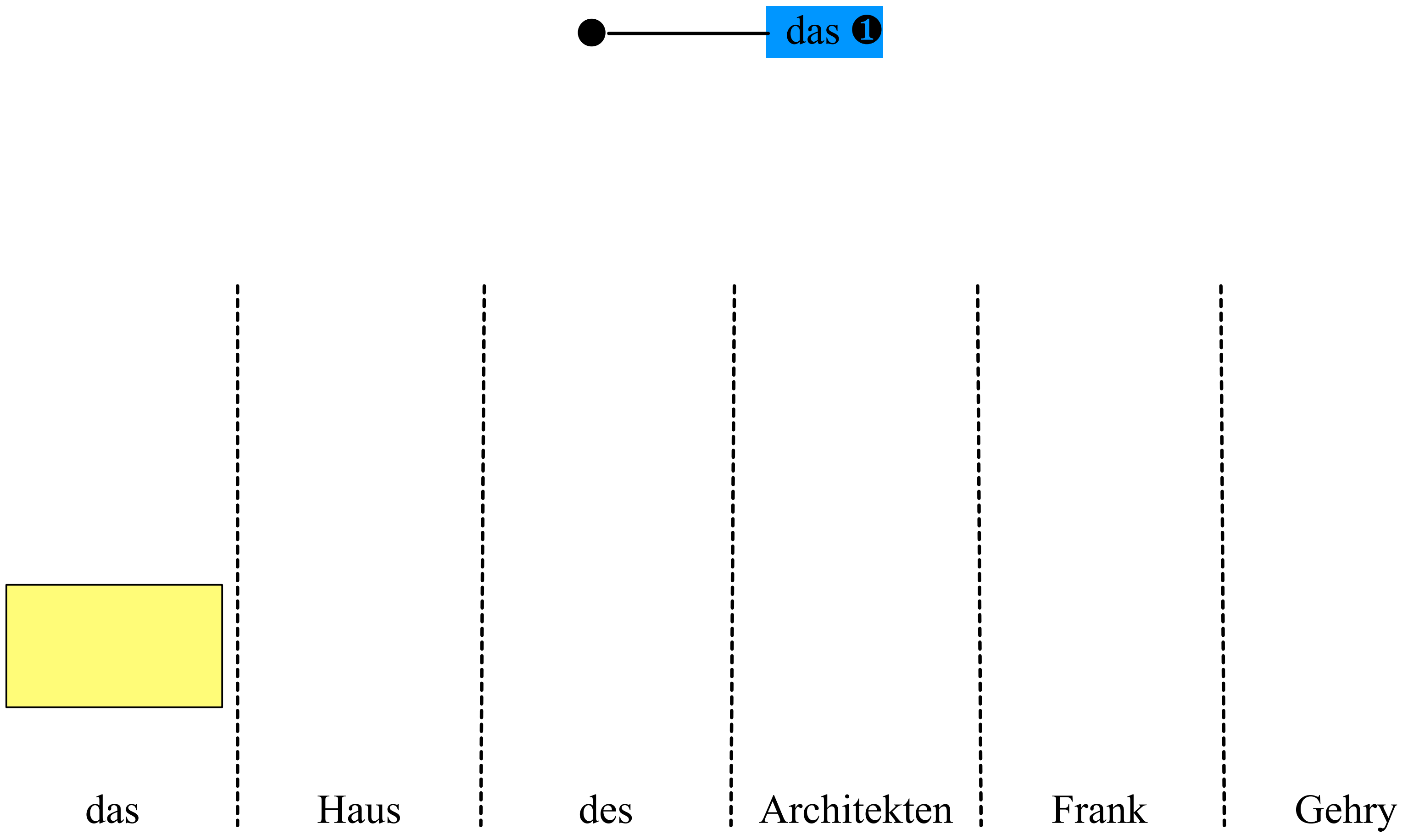
Finding Applicable Rules in Prefix Tree



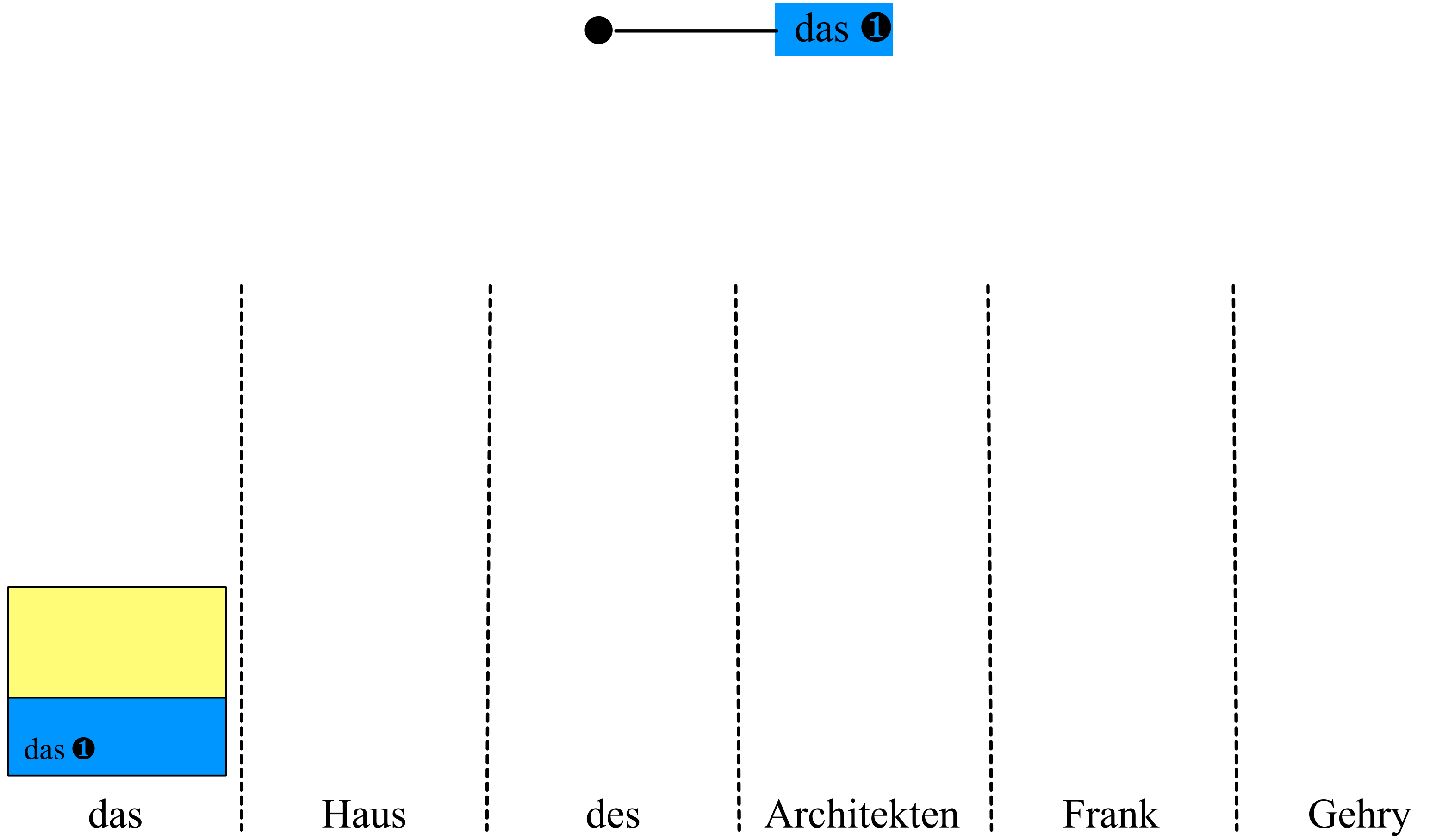
Covering the First Cell



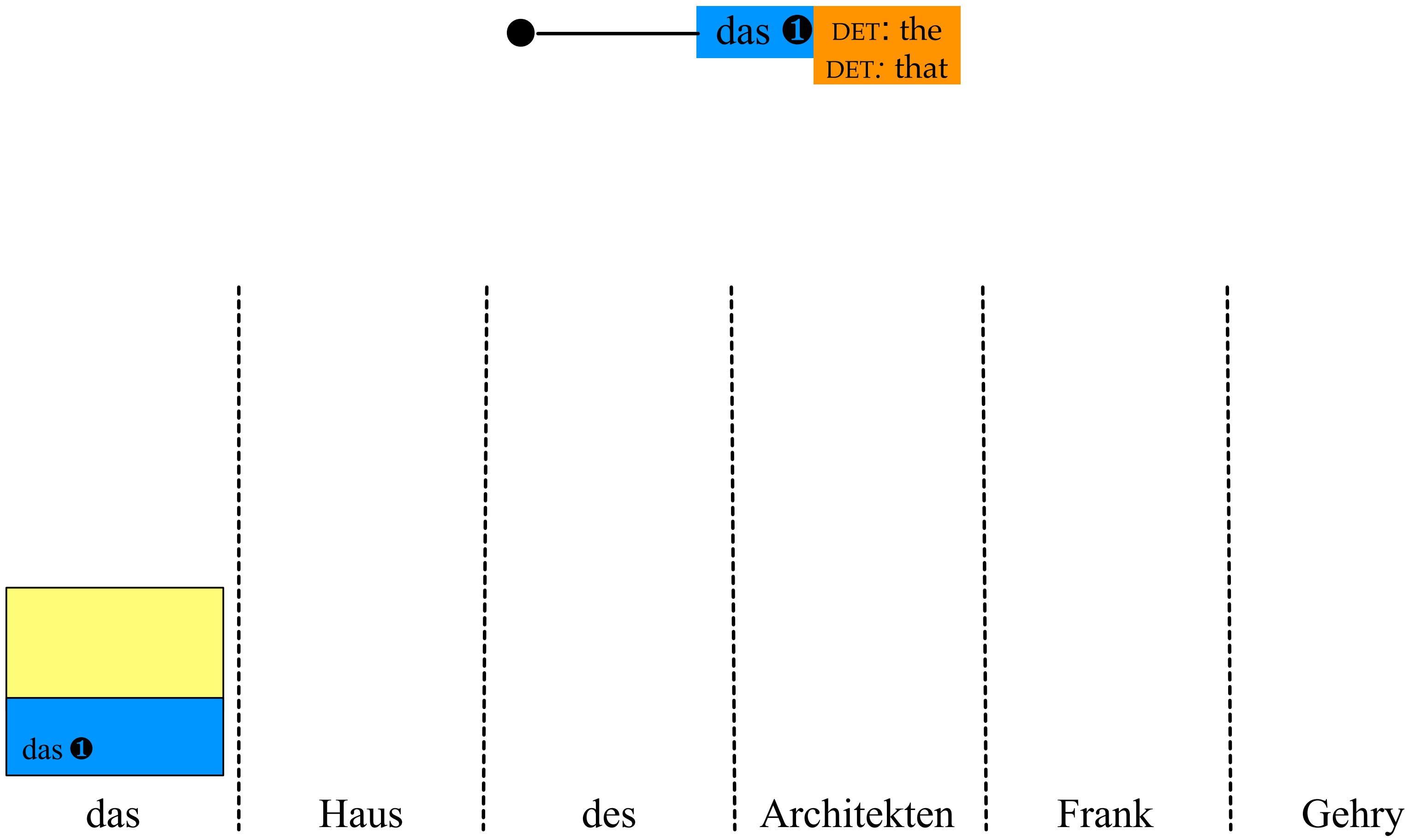
Looking up Rules in the Prefix Tree



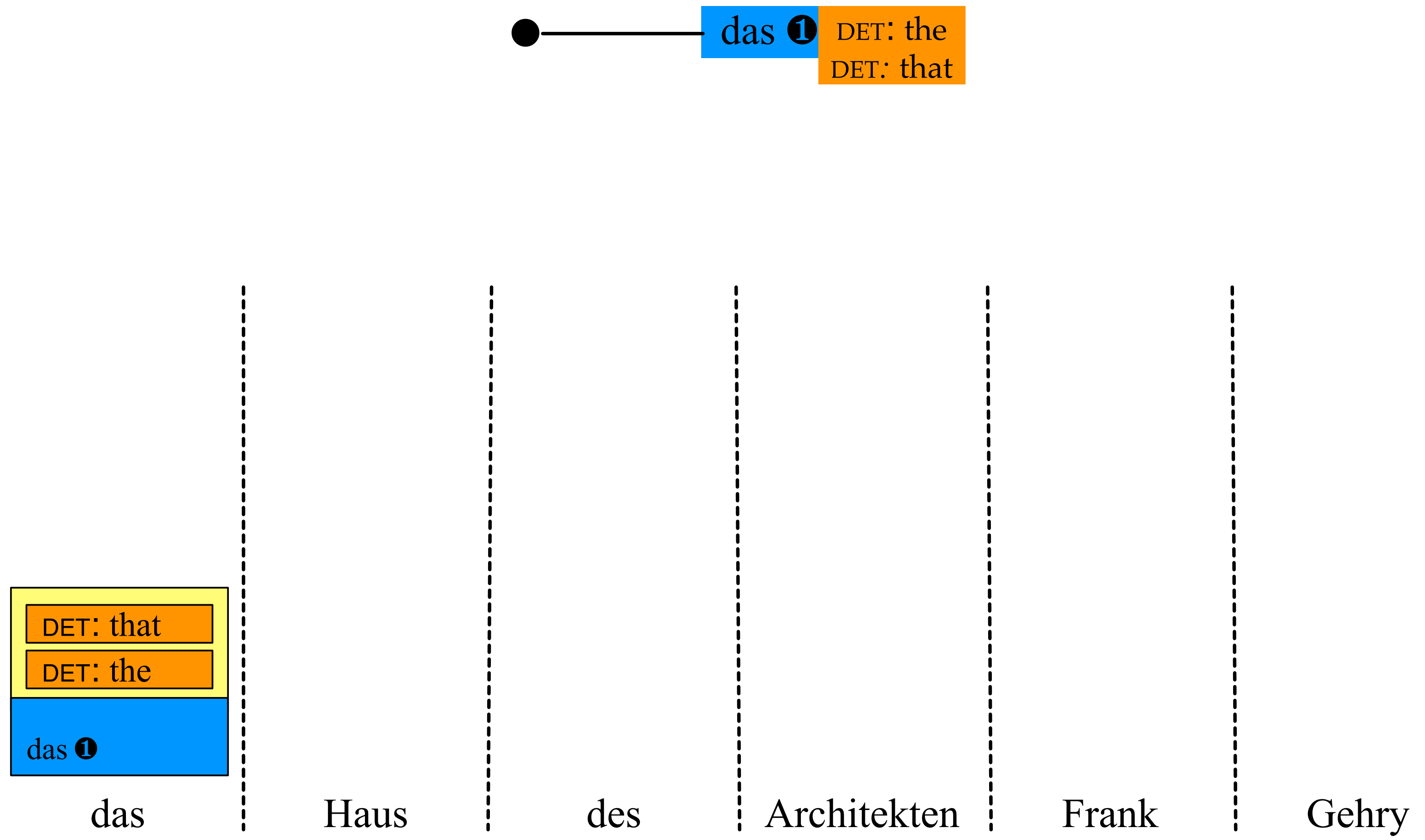
Taking Note of the Dotted Rule



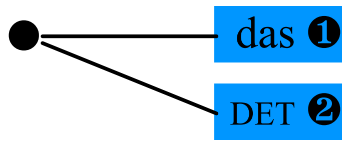
Checking if Dotted Rule has Translations

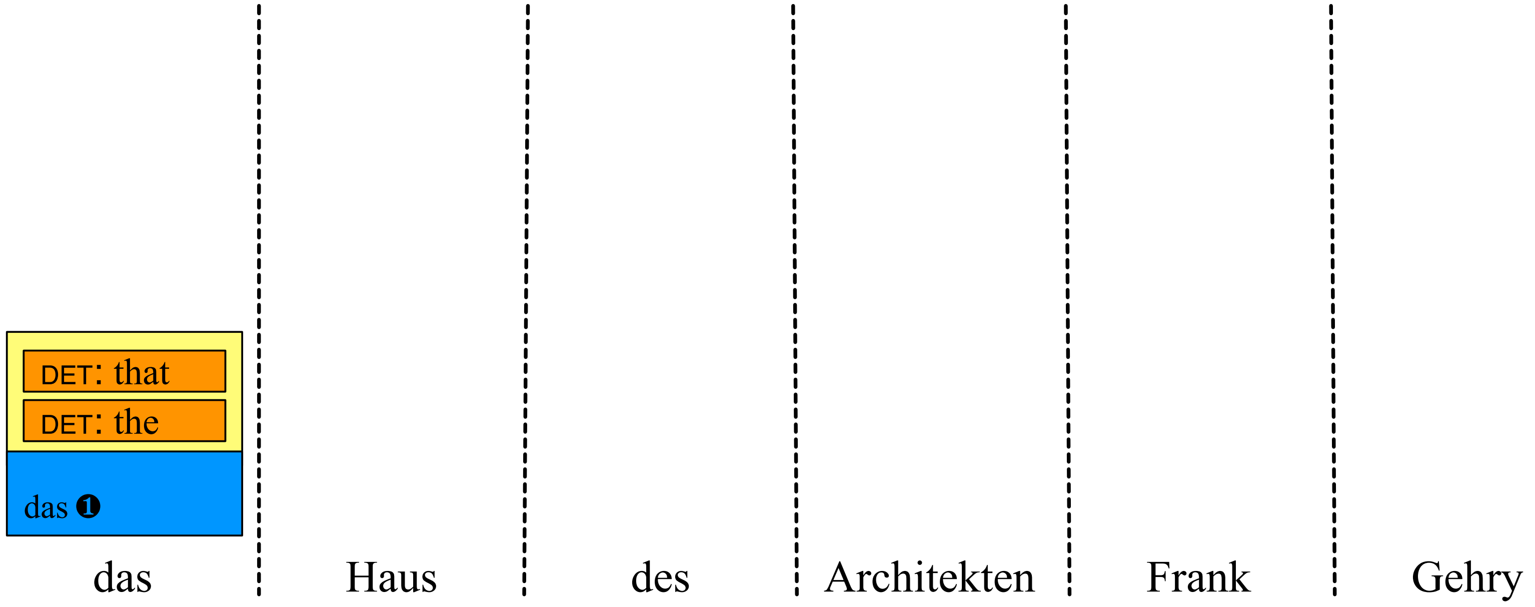


Applying the Translation Rules

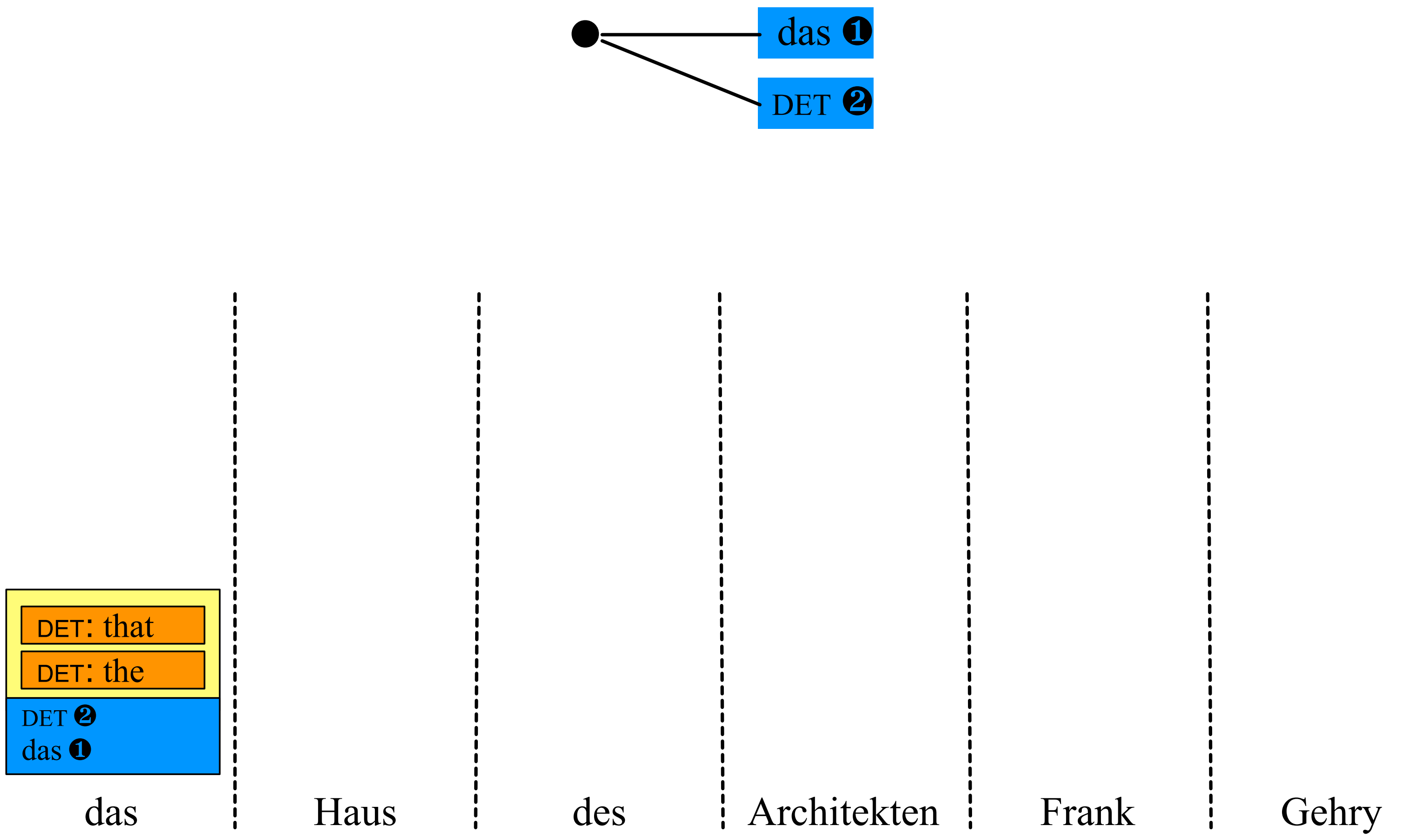


Looking up Constituent Label in Prefix Tree

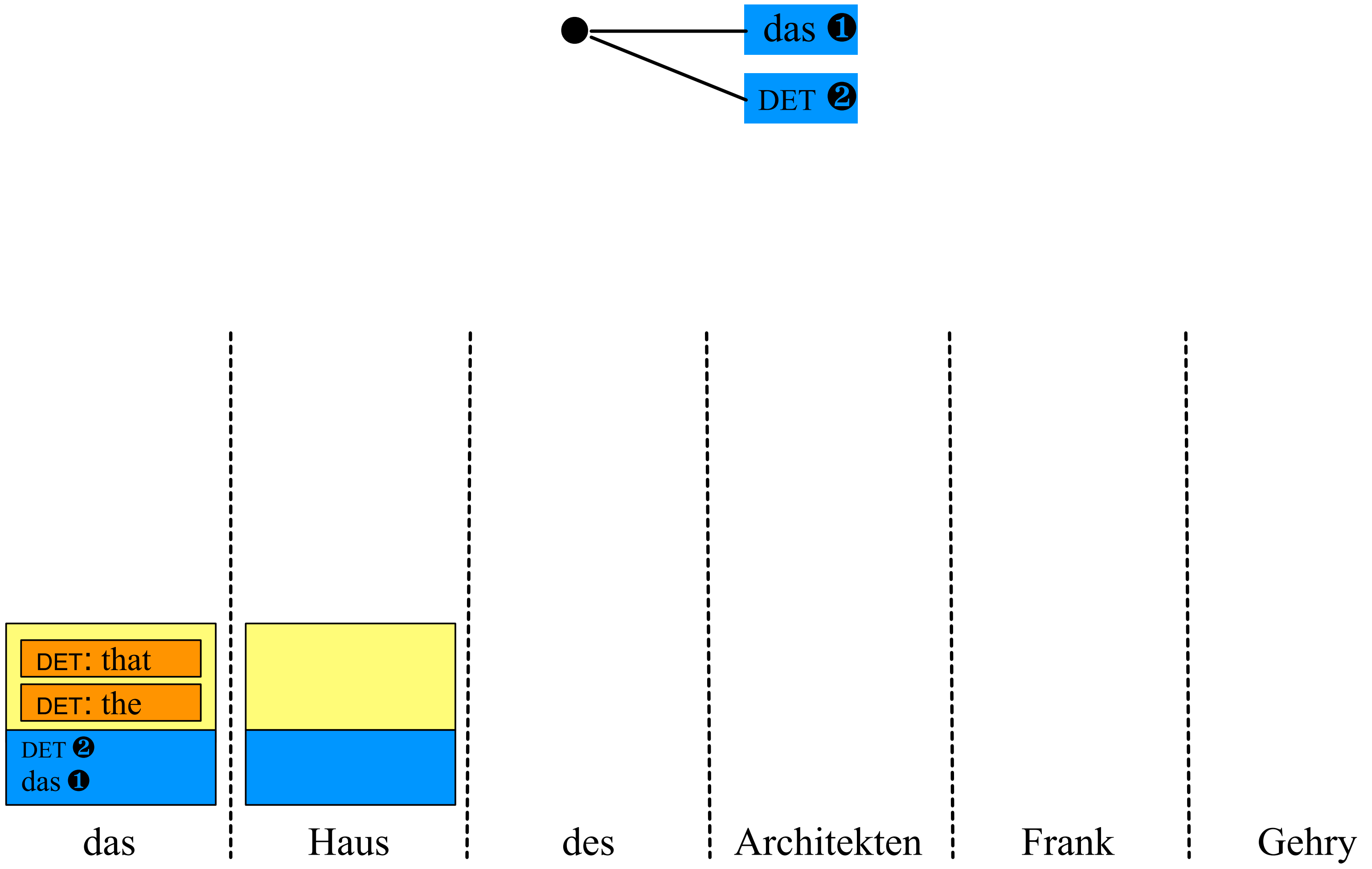




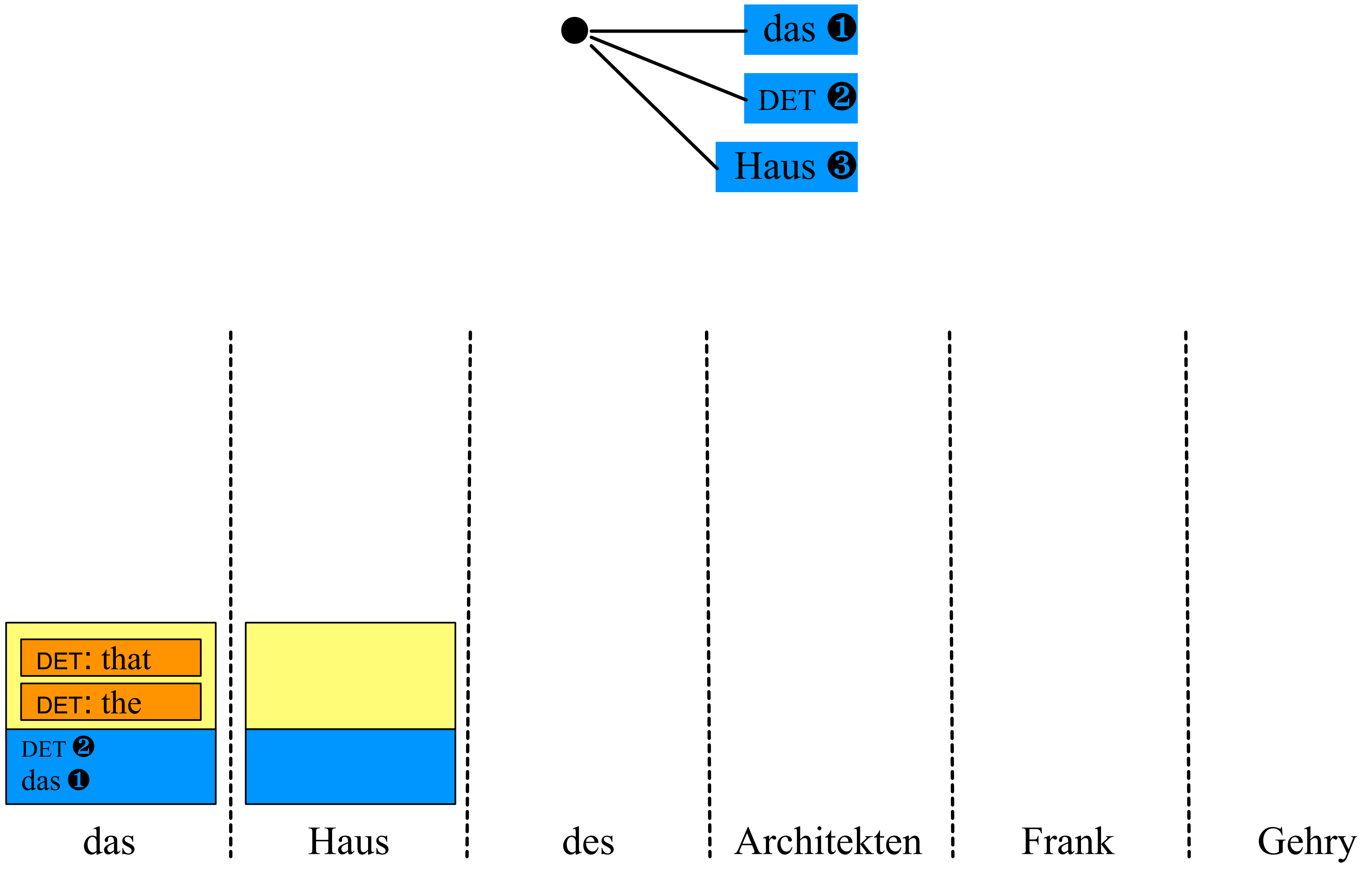
Add to Span's List of Dotted Rules



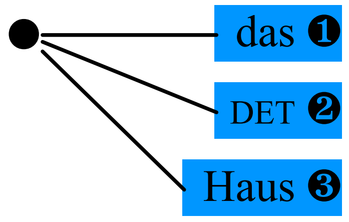
Moving on to the Next Cell

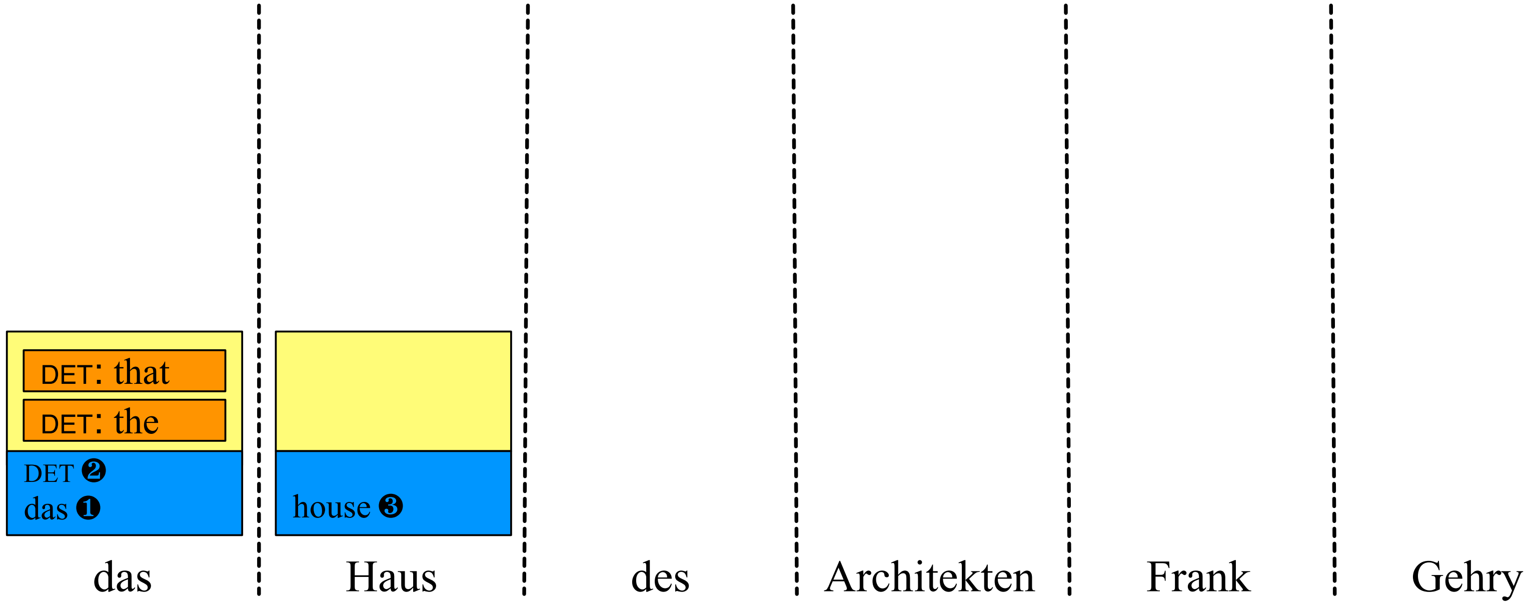


Looking up Rules in the Prefix Tree

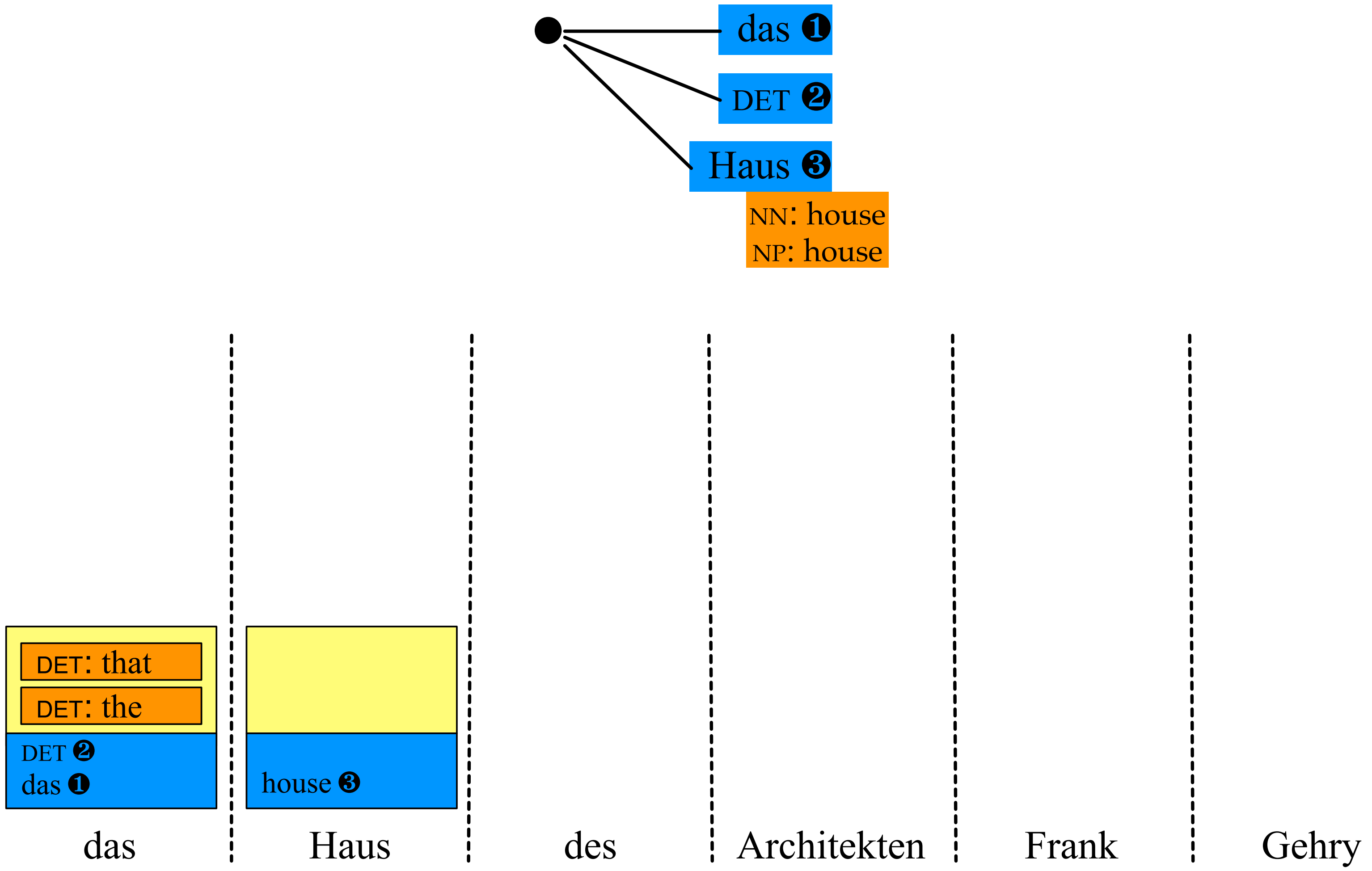


Taking Note of the Dotted Rule

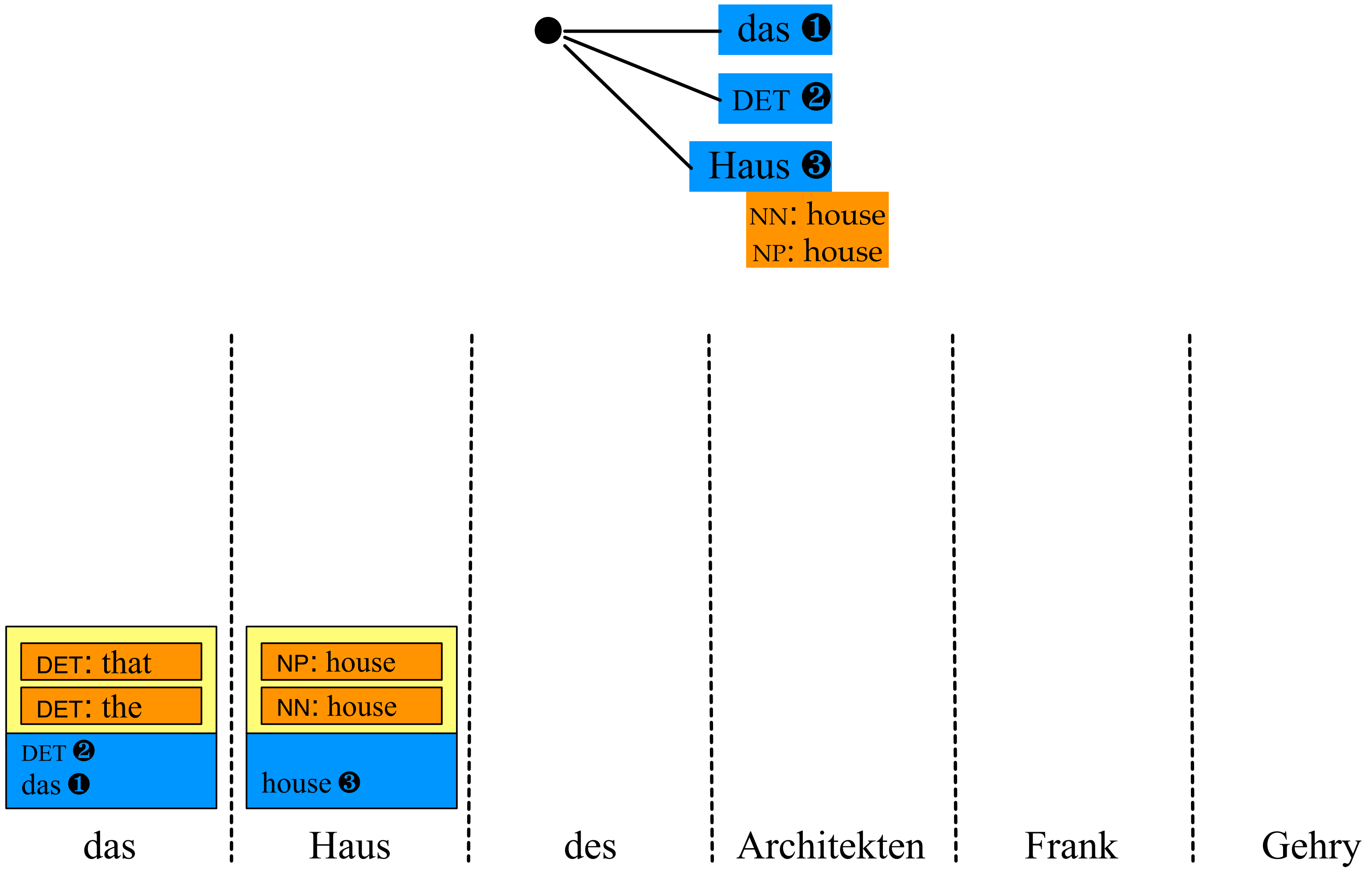




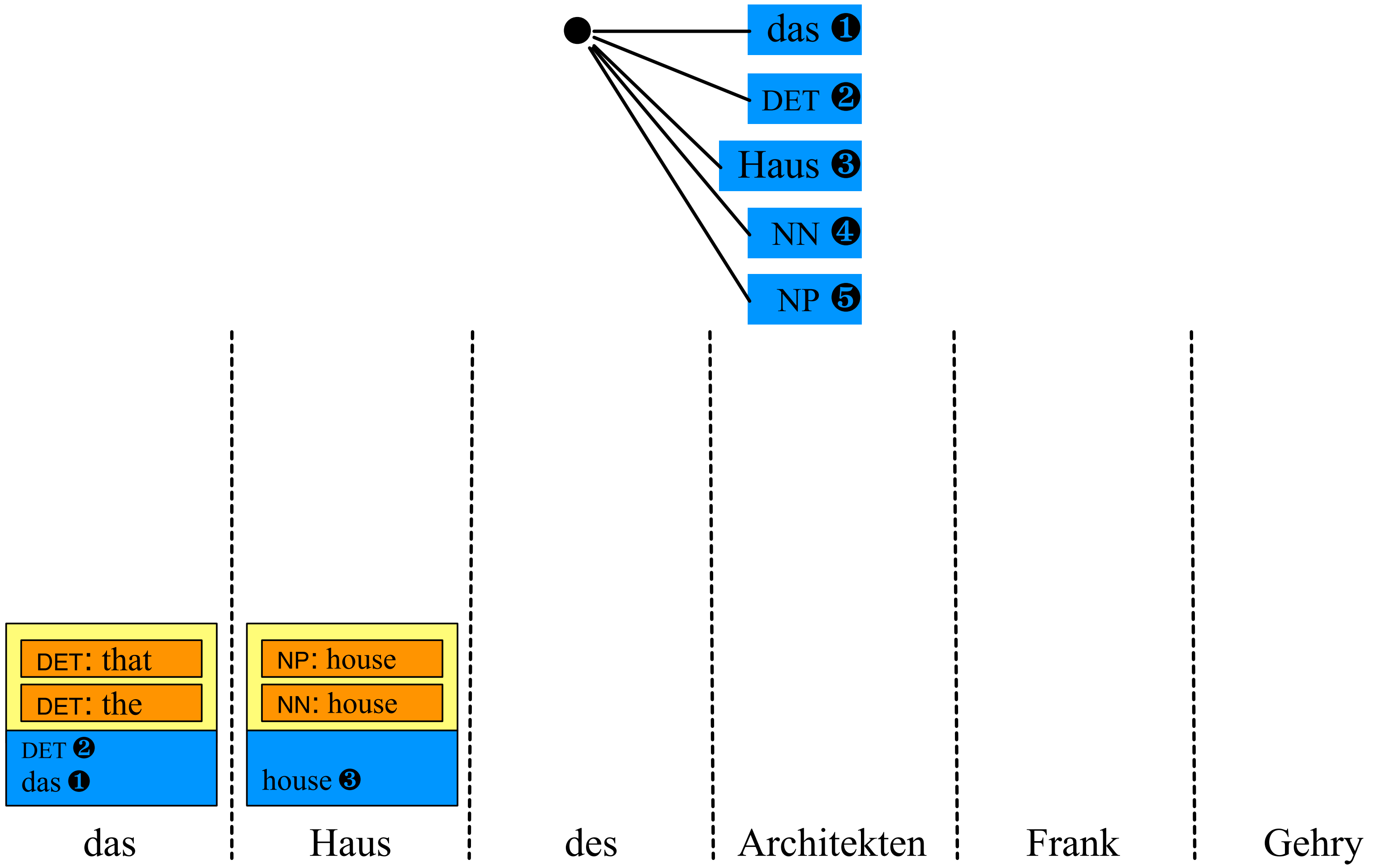
Checking if Dotted Rule has Translations



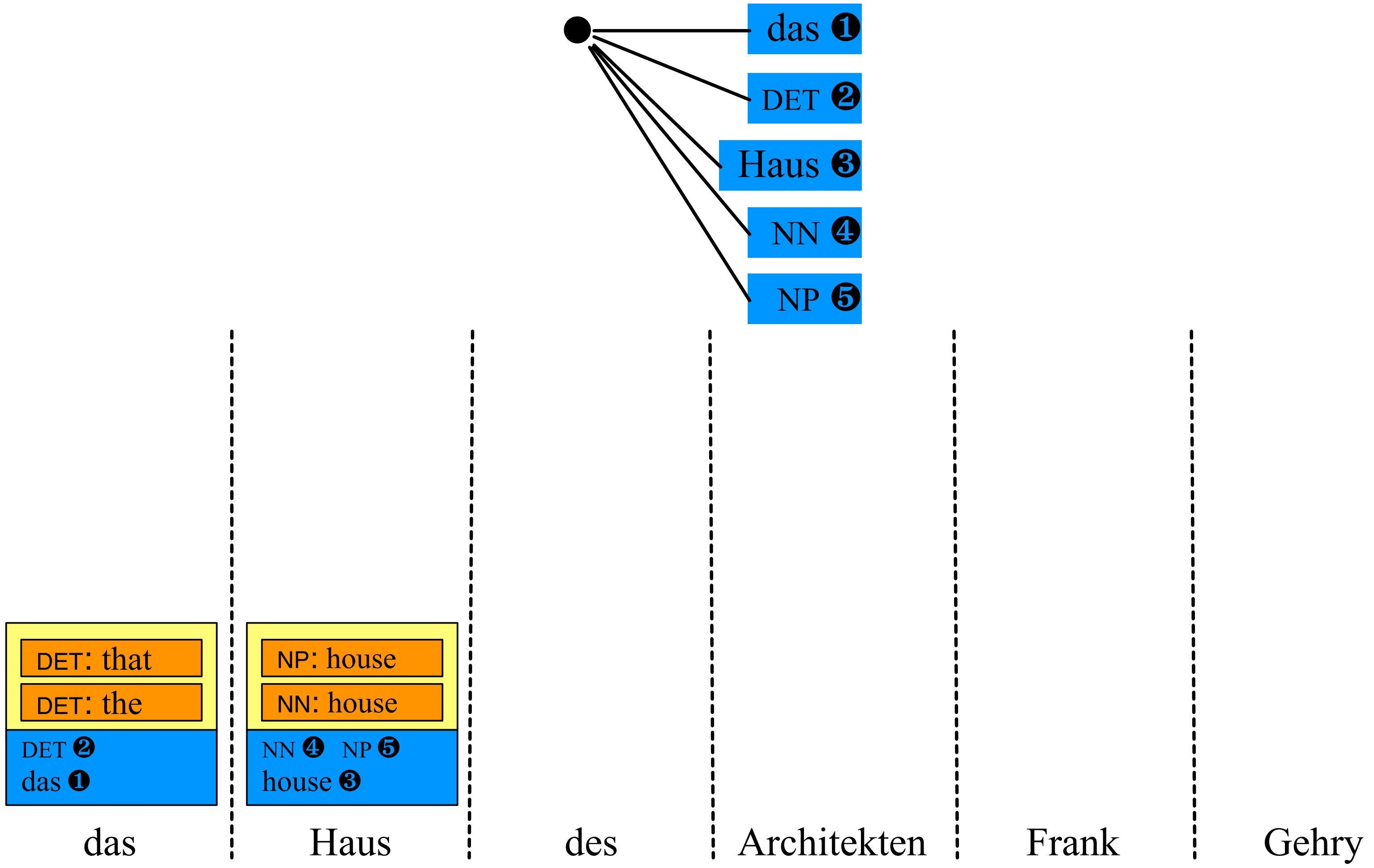
Applying the Translation Rules



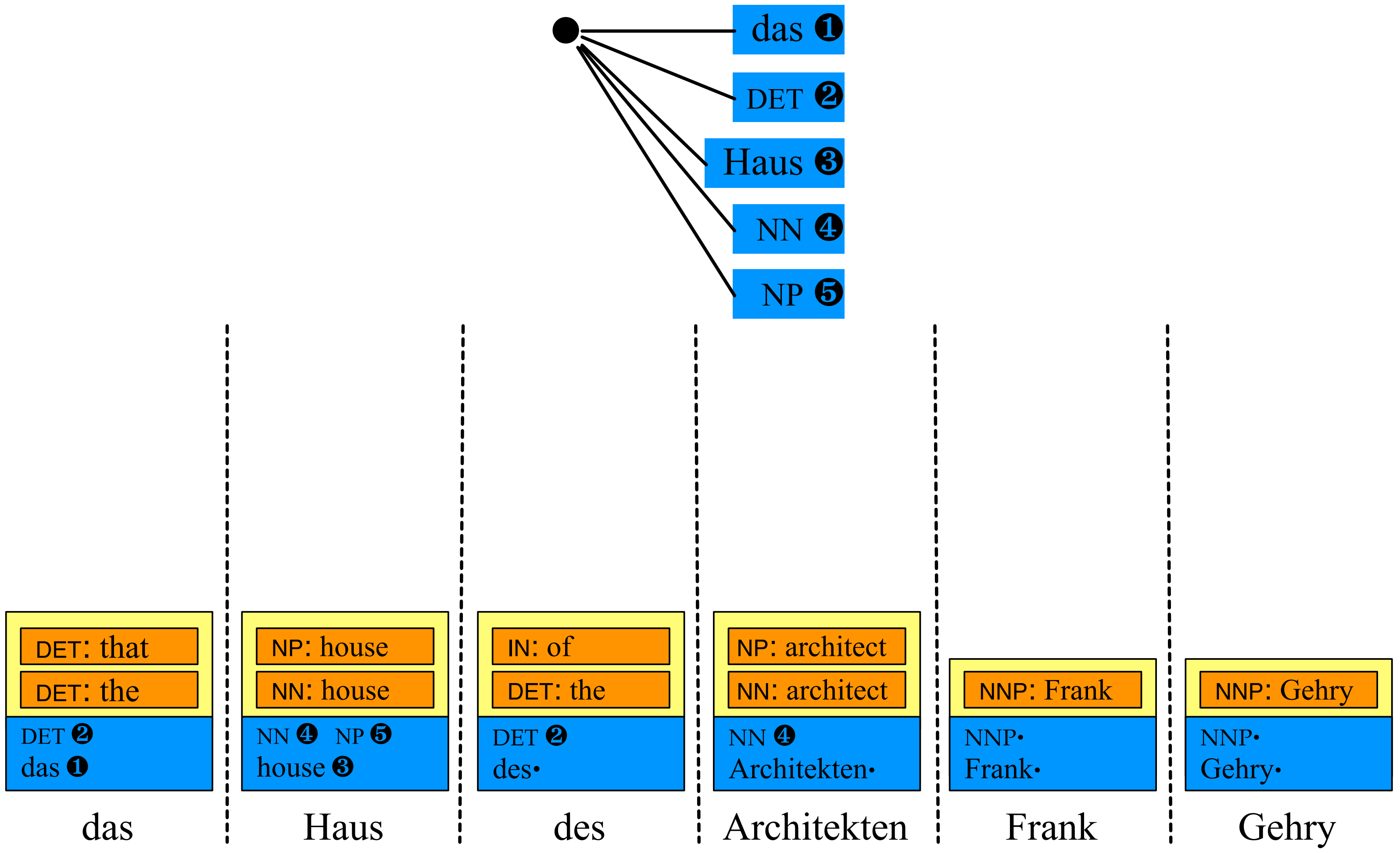
Looking up Constituent Label in Prefix Tree



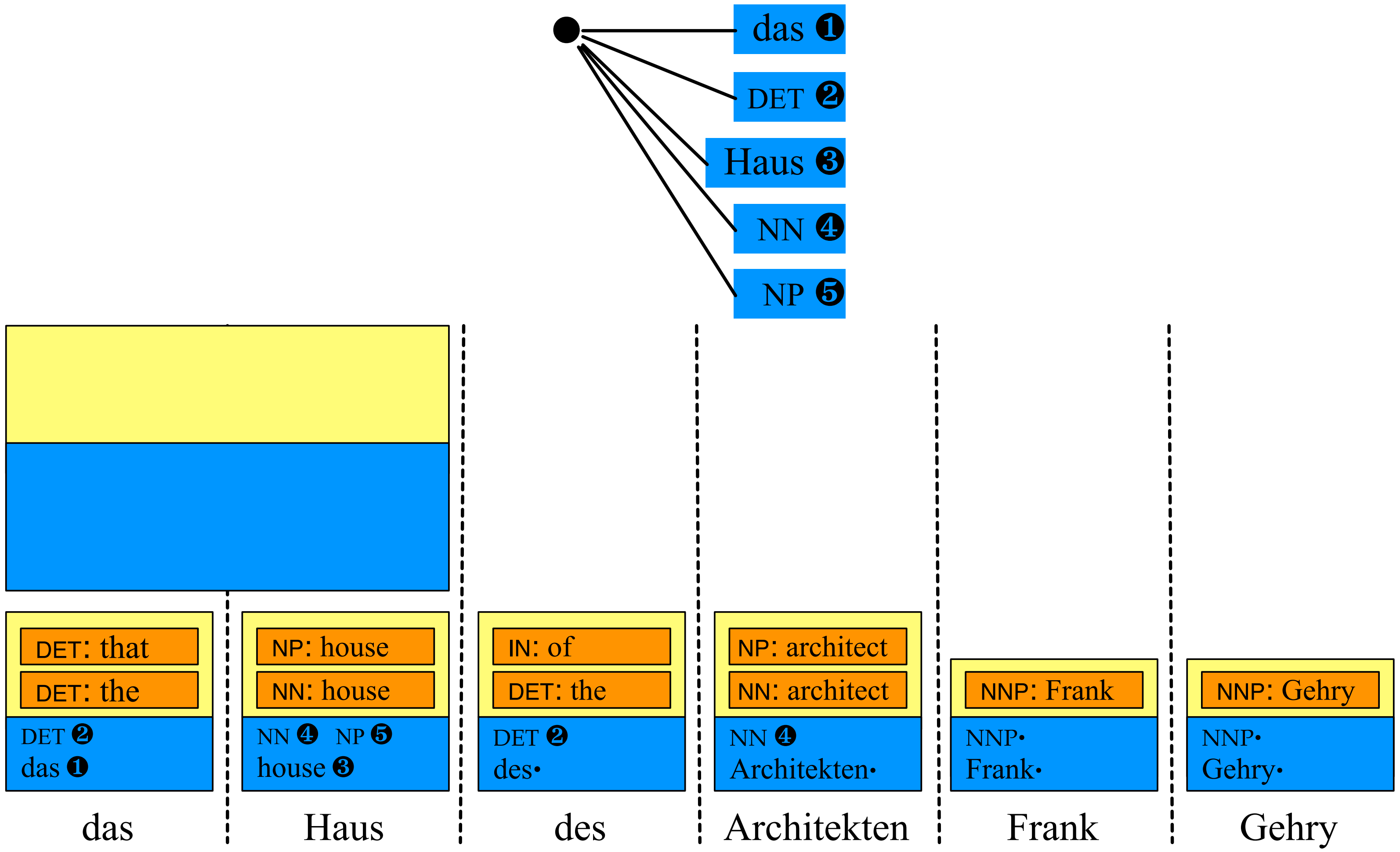
Add to Span’s List of Dotted Rules



More of the Same



Moving on to the Next Cell

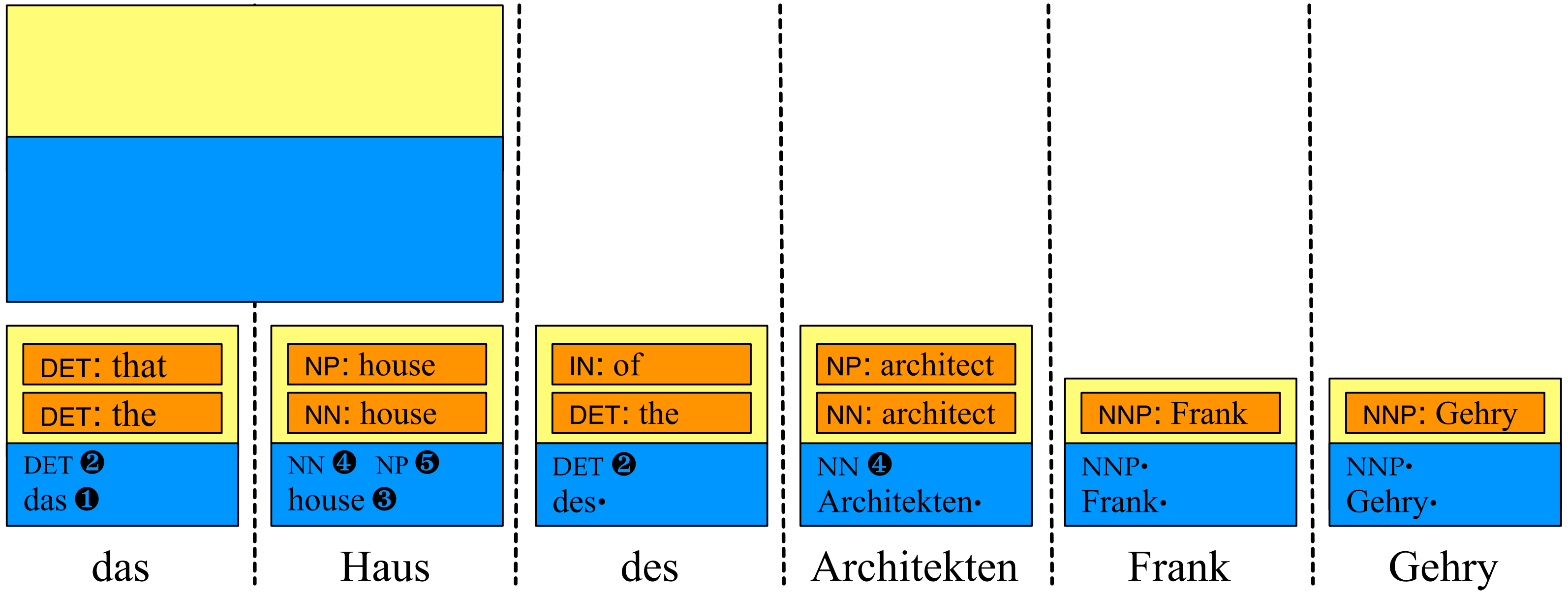


Covering a Longer Span

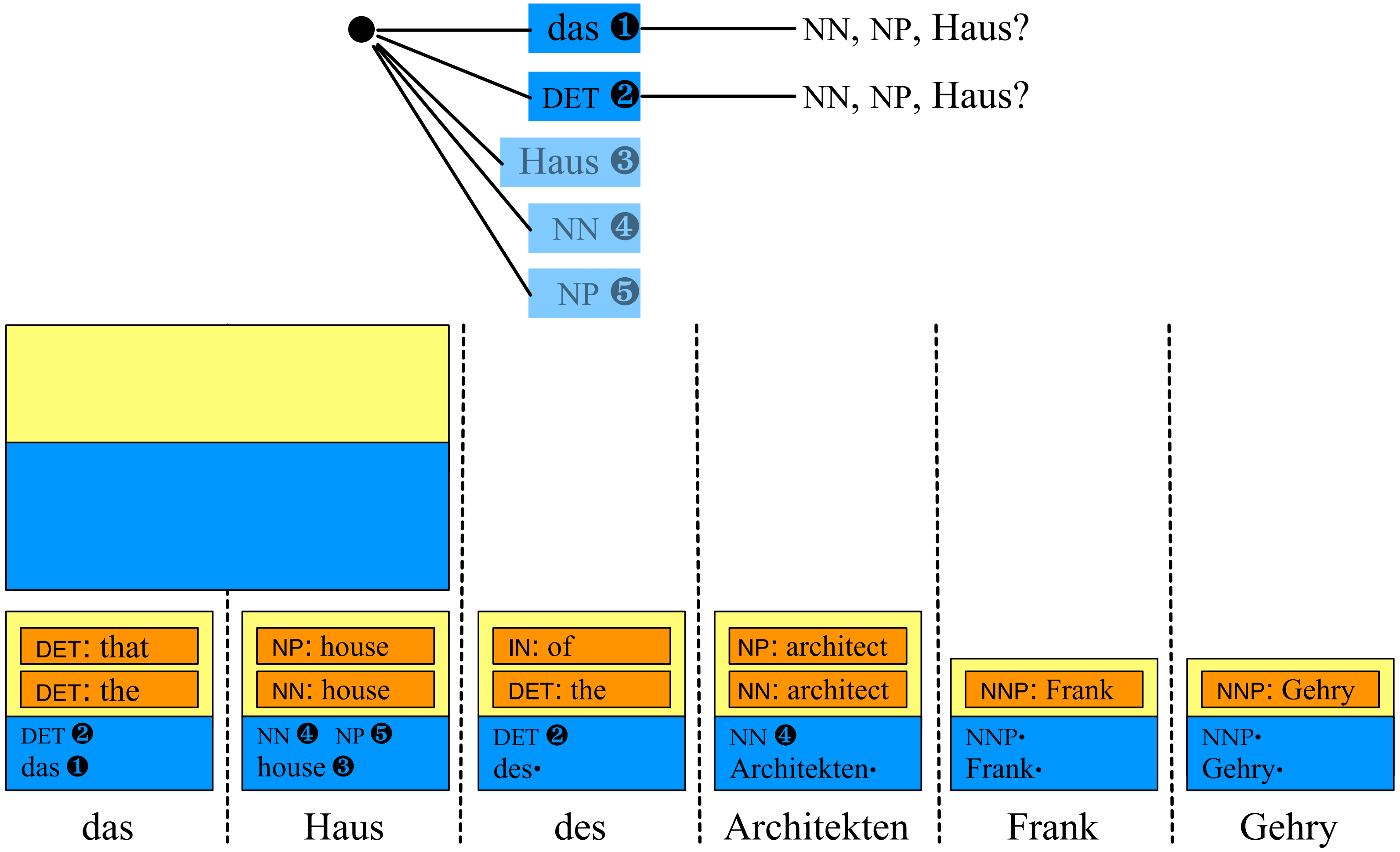
Cannot consume multiple words at once

All rules are extensions of existing dotted rules

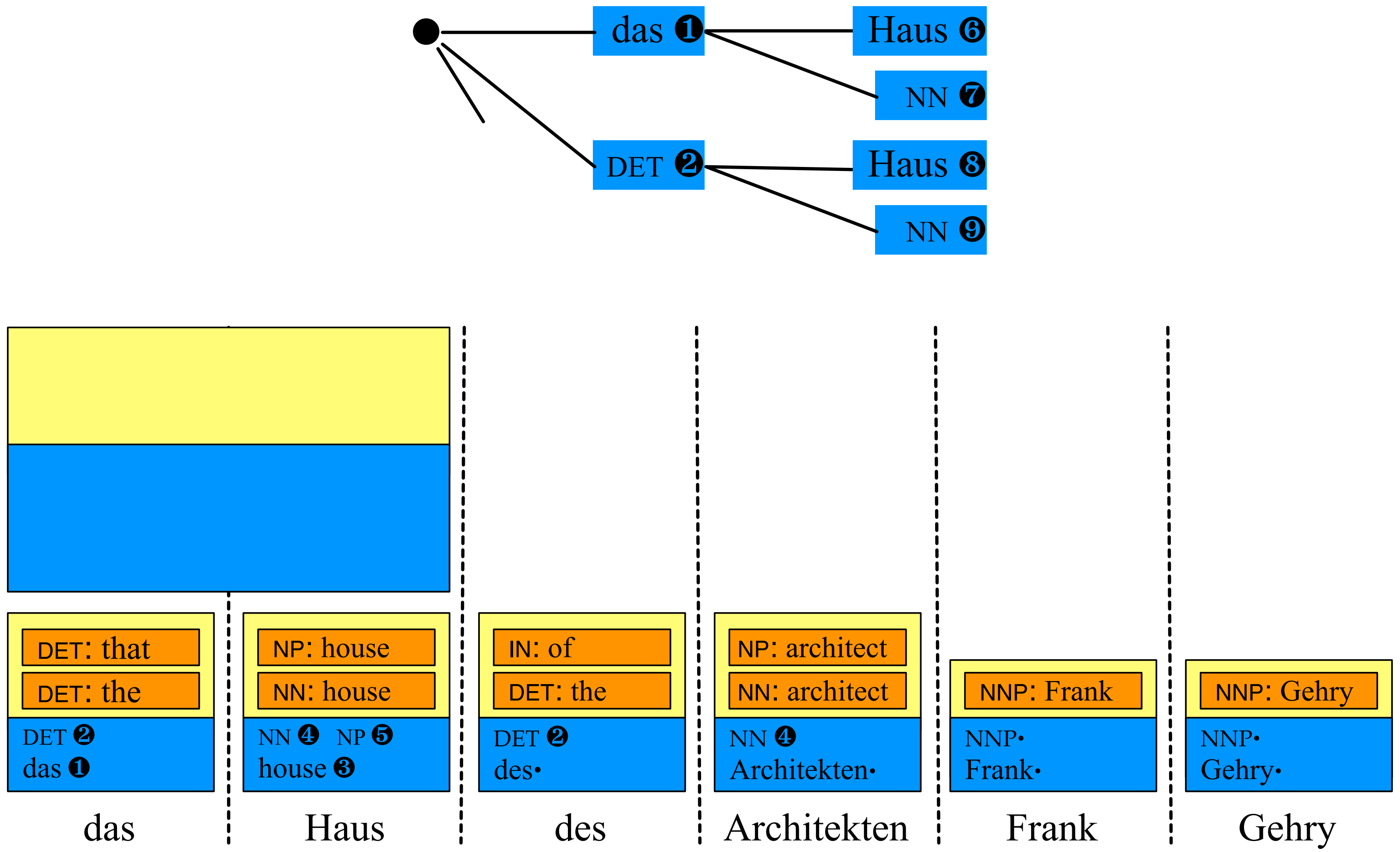
Here: only extensions of span over das possible



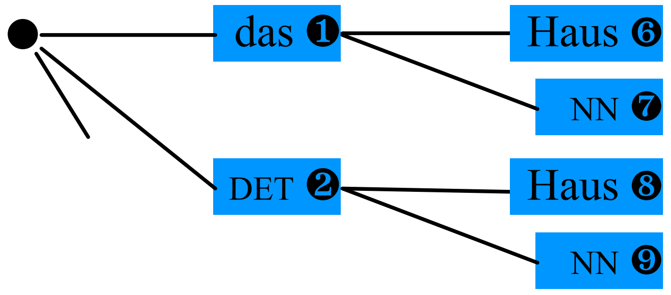
Extensions of Span over das

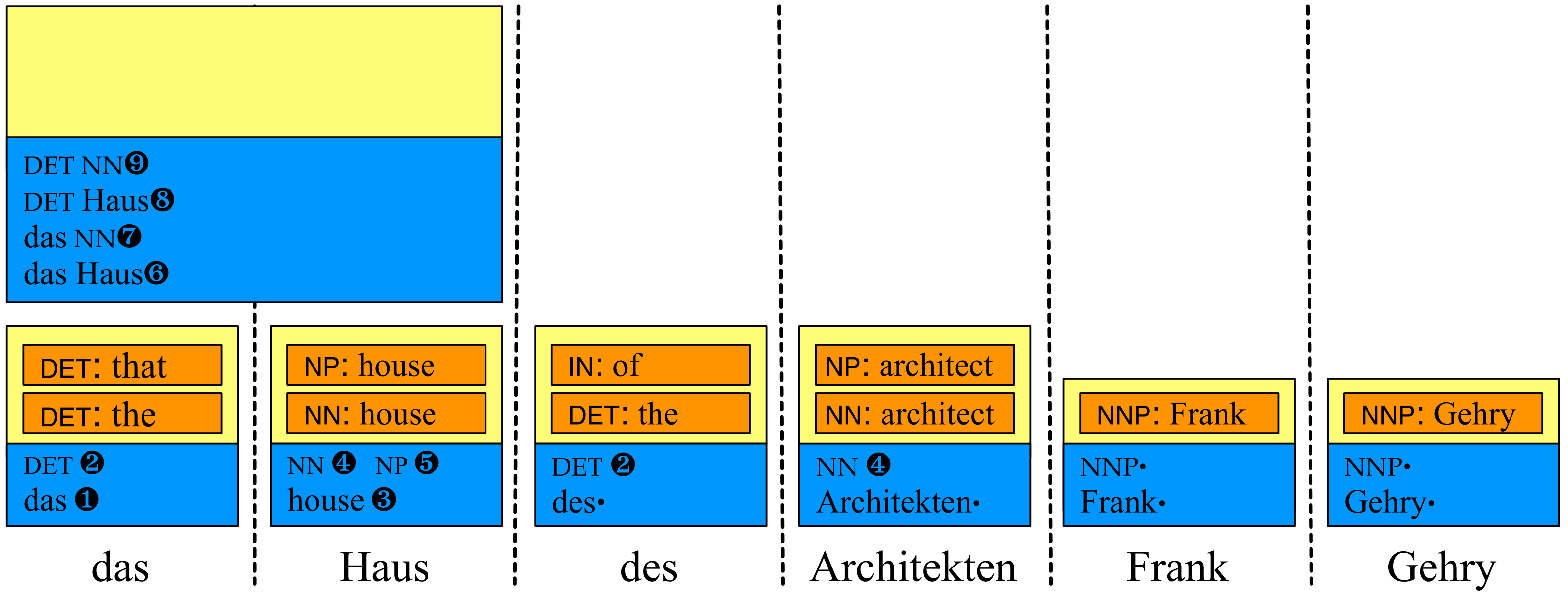


Looking up Rules in the Prefix Tree

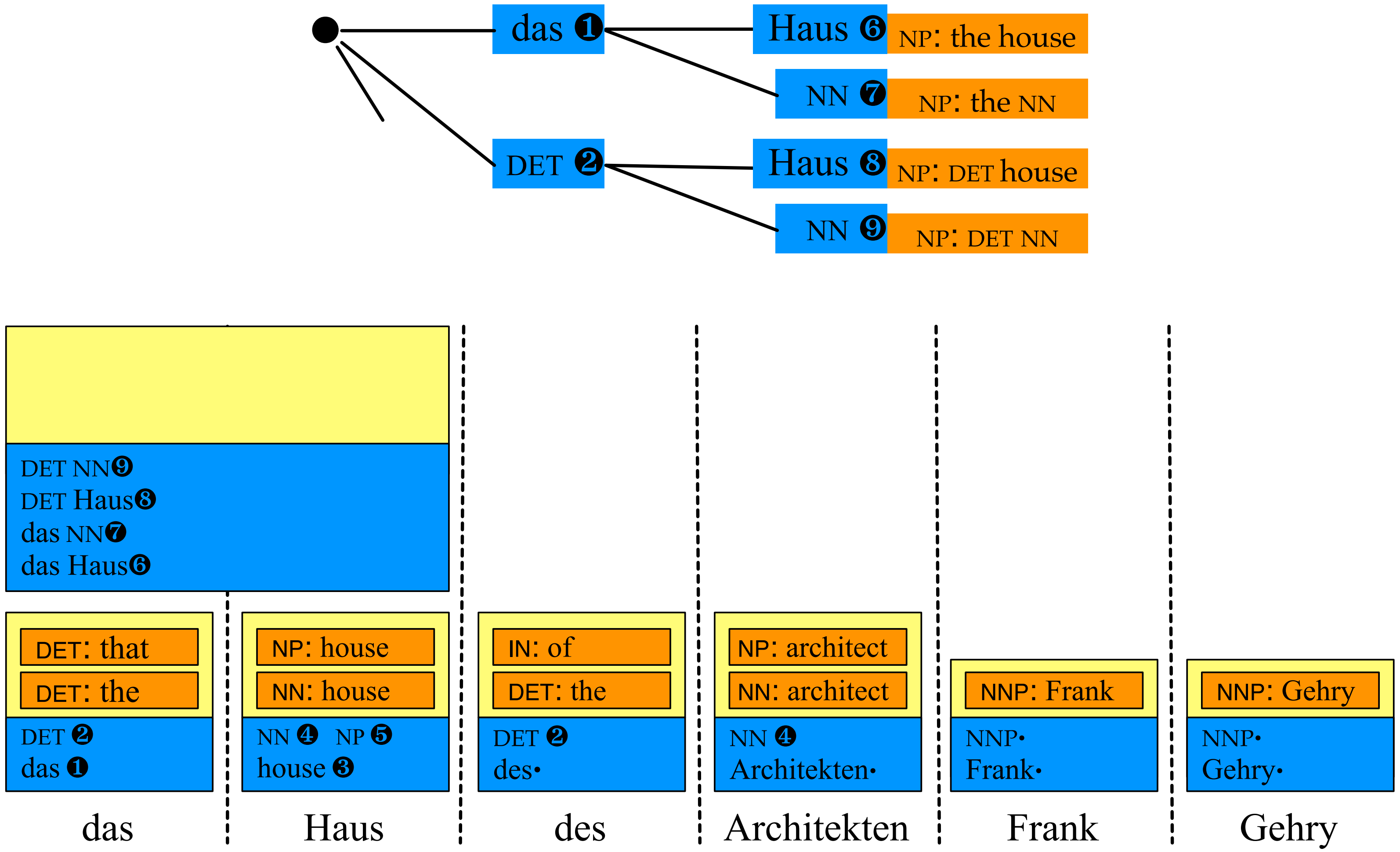


Taking Note of the Dotted Rule

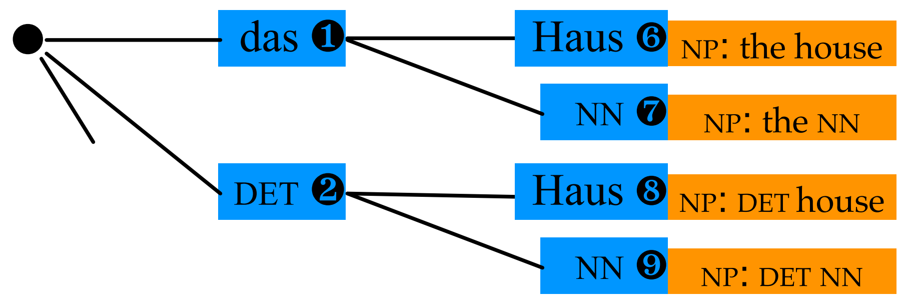


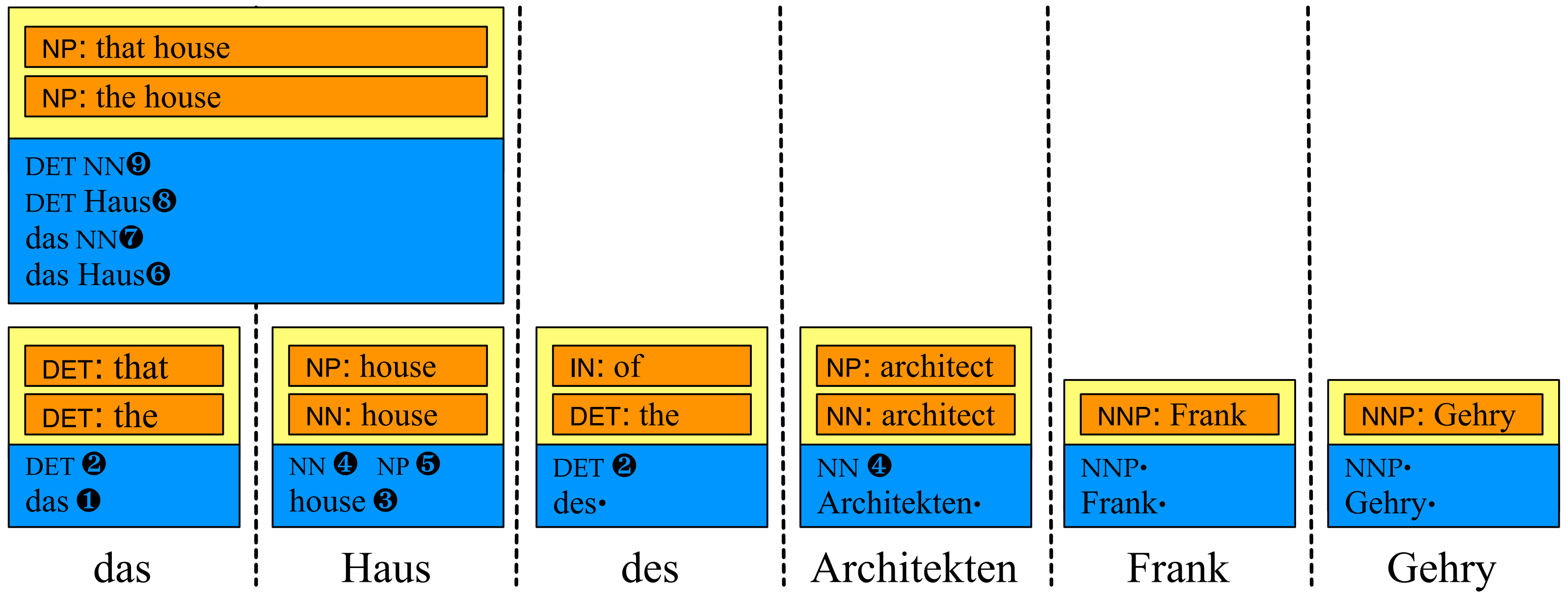


Checking if Dotted Rules have Translations

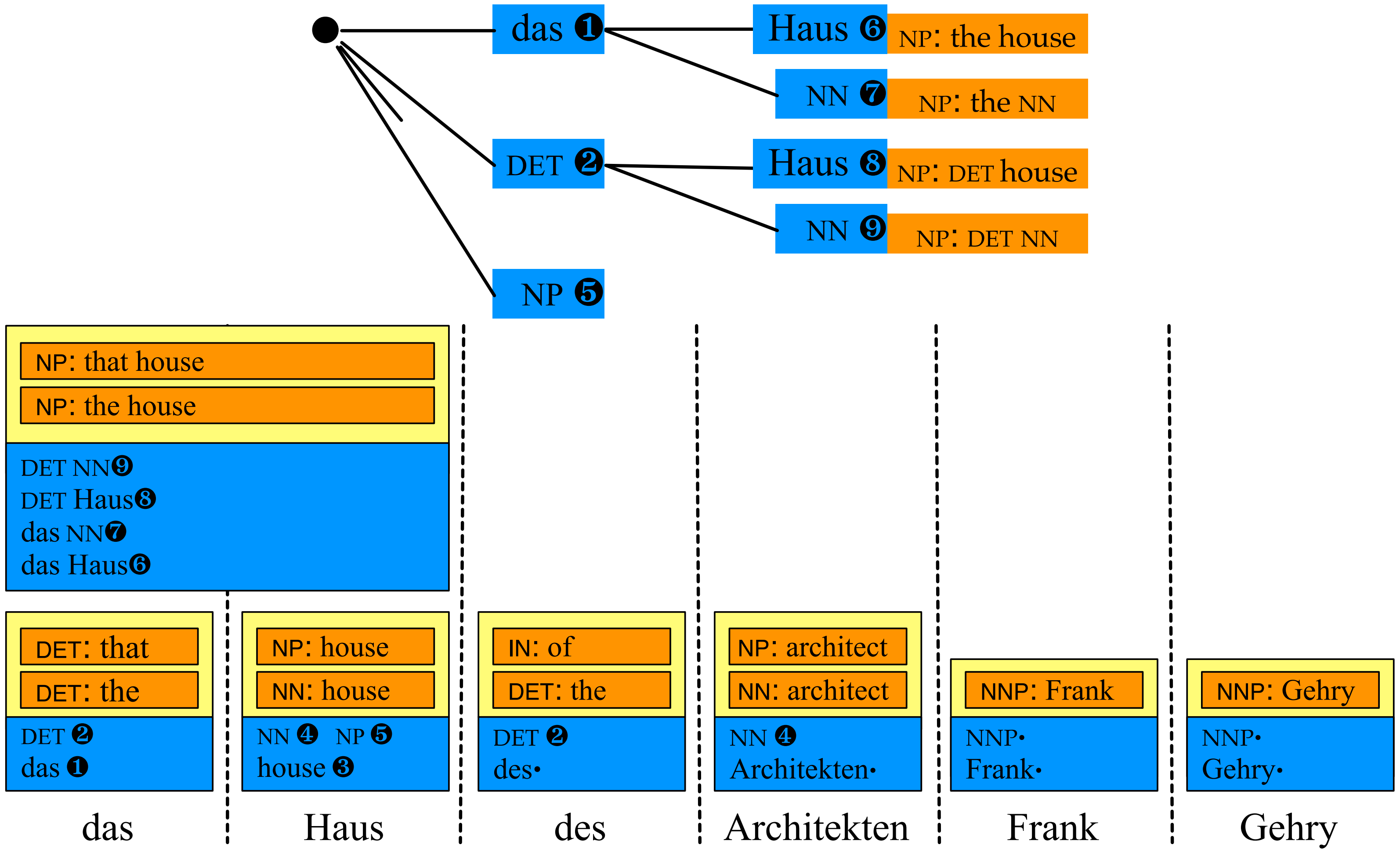


Applying the Translation Rules

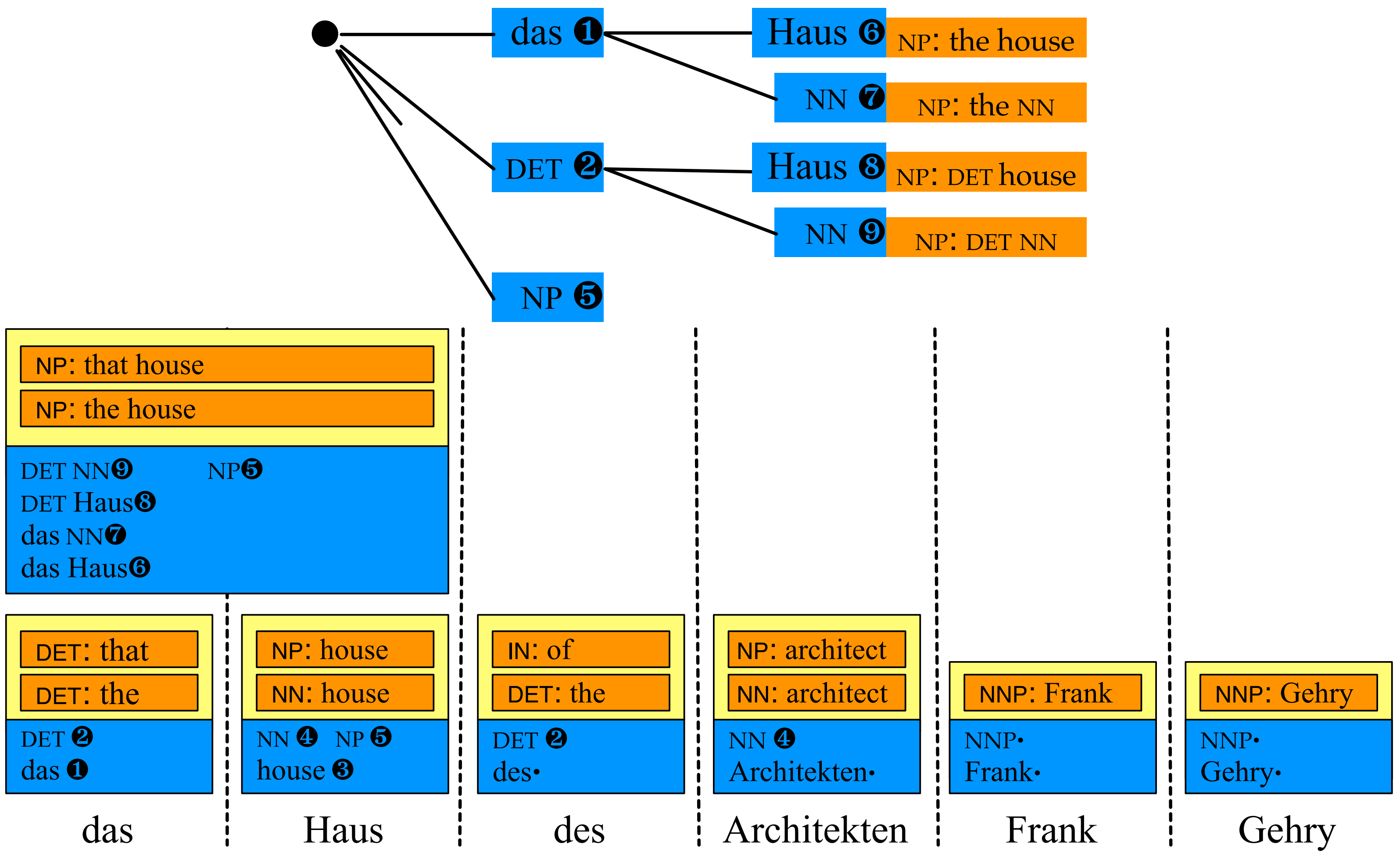




Looking up Constituent Label in Prefix Tree



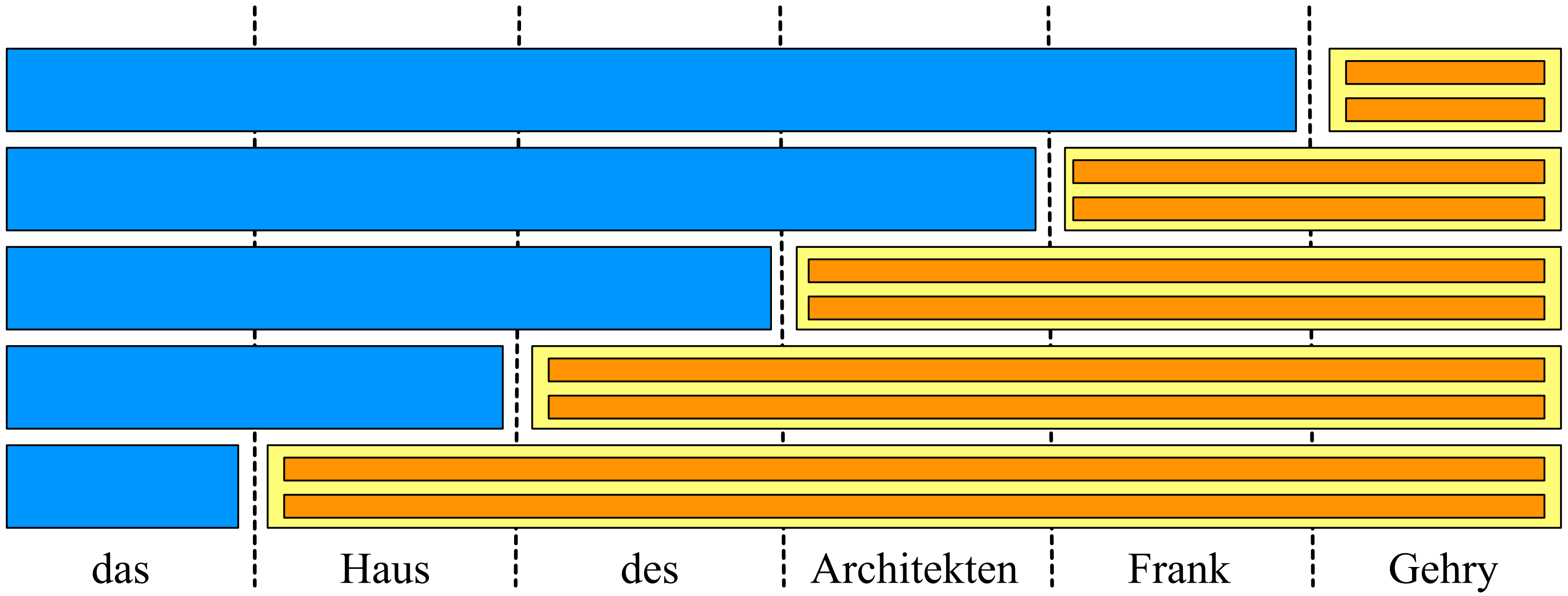
Add to Span's List of Dotted Rules



Even Larger Spans

Extend lists of dotted rules with cell constituent labels

span's dotted rule list (with same start)
plus neighboring
span's constituent labels of hypotheses (with same end)



Reflections

- Complexity $O(rn^3)$ with sentence length n and size of dotted rule list r
  - may introduce maximum size for spans that do not start at beginning
  - may limit size of dotted rule list (very arbitrary)

- Does the list of dotted rules explode?

- Yes, if there are many rules with neighboring target-side non-terminals
  - such rules apply in many places
  - rules with words are much more restricted

Difficult Rules

- Some rules may apply in too many ways

- Neighboring input non-terminals

$$\text{NP} \rightarrow \text{X}_1\ \text{X}_2 \mid \text{NP}_2 \text{ to } \text{NP}_1$$

  - non-terminals may match many different pairs of spans
  - especially a problem for hierarchical models (no constituent label restrictions)
  - may be okay for syntax-models

- Three neighboring input non-terminals

$$\text{VP} \rightarrow \text{trifft } \text{X}_1\ \text{X}_2\ \text{X}_3 \text{ heute} \mid \text{meets } \text{NP}_1 \text{ today } \text{PP}_2\ \text{PP}_3$$

  - will get out of hand even for syntax models

Summary

- Basic idea: bottom up chart parsing

- Prefix structure for easy rule access

- Caching rule matching with dotted rules

- Coming up...
  - cube pruning for syntax-based decoding
  - recombination and state
  - scope3 pruning
  - recursive cky+
  - coarse-to-fine