Chart-Based Decoding

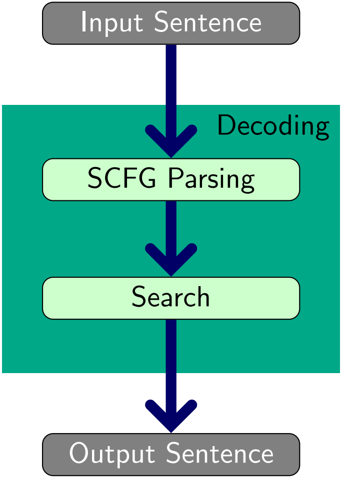
Kenneth Heafield

University of Edinburgh

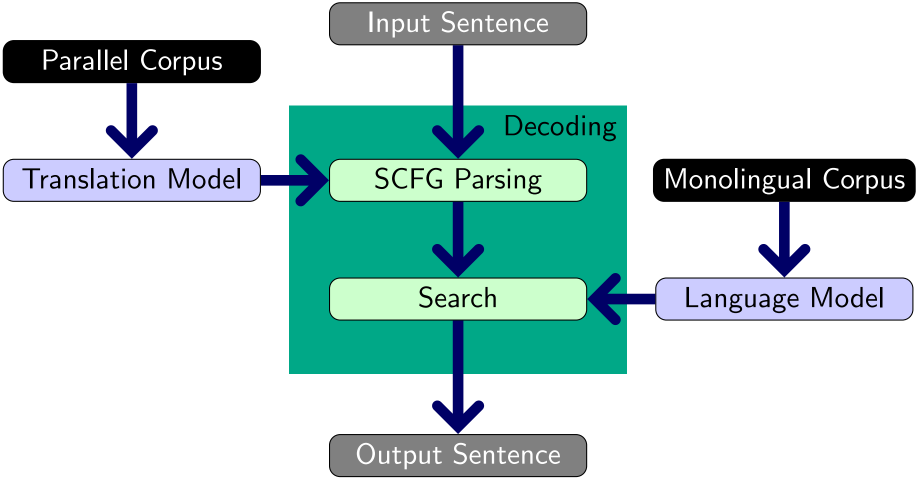
6 September, 2012

Most slides courtesy of Philipp Koehn

Overview of Syntactic Decoding



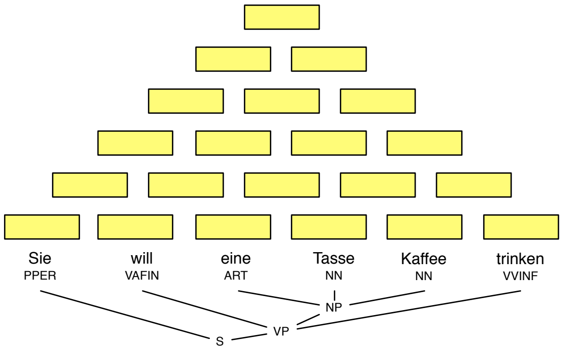
Overview of Syntactic Decoding



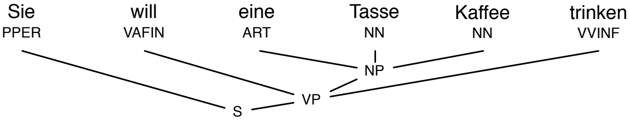
Syntactic Decoding

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

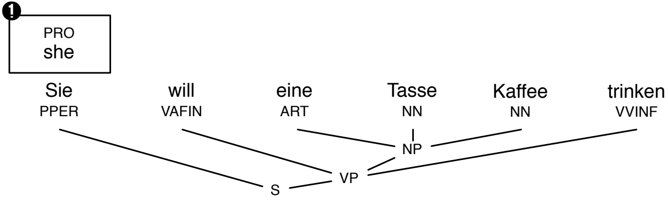


Syntax Decoding



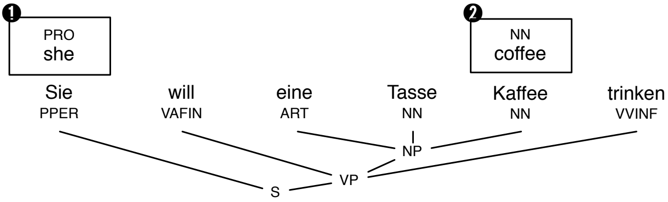
German input sentence with tree

Syntax Decoding



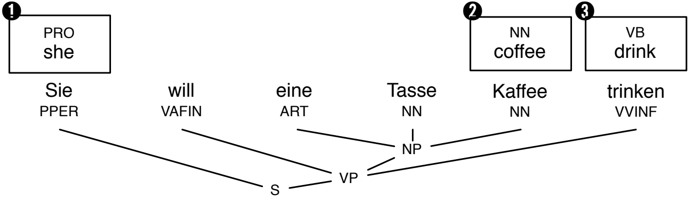
Purely lexical rule: filling a span with a translation (a constituent)

Syntax Decoding



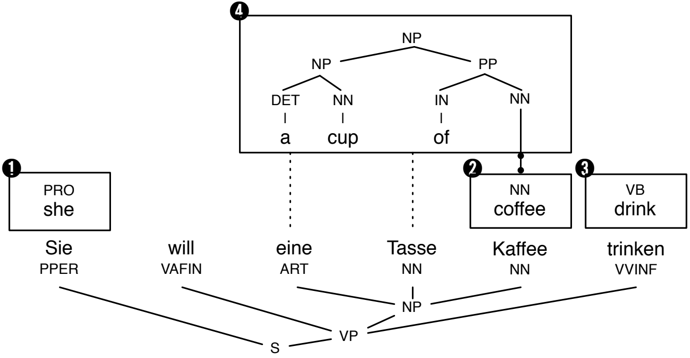
Purely lexical rule: filling a span with a translation (a constituent)

Syntax Decoding



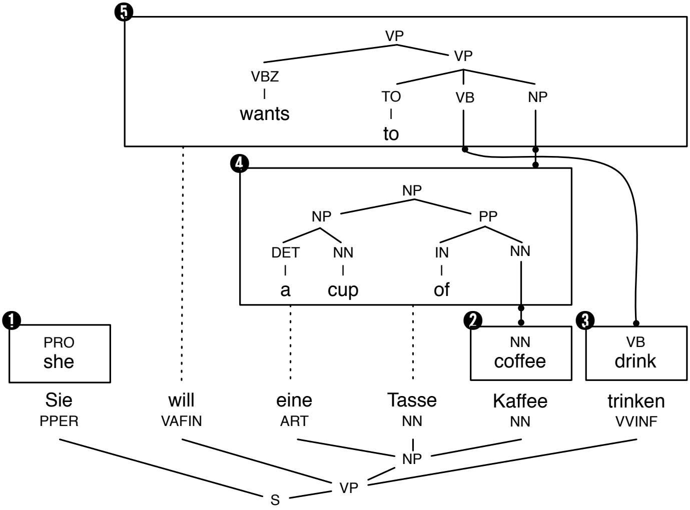
Purely lexical rule: filling a span with a translation (a constituent)

Syntax Decoding



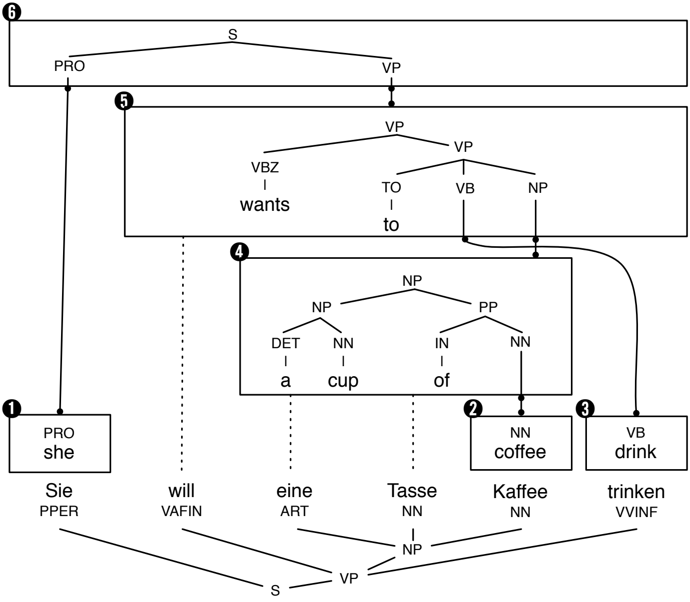
Complex rule: matching underlying constituent spans, and covering words

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

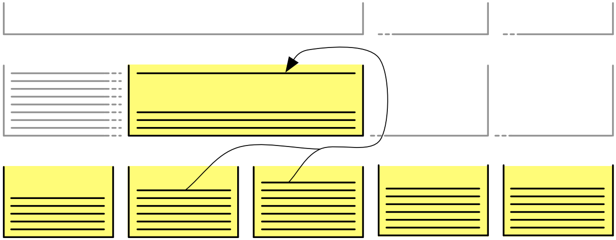
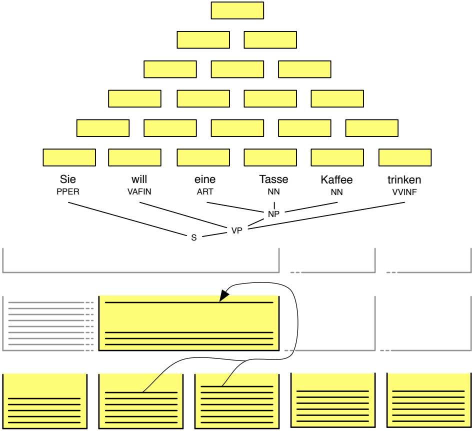


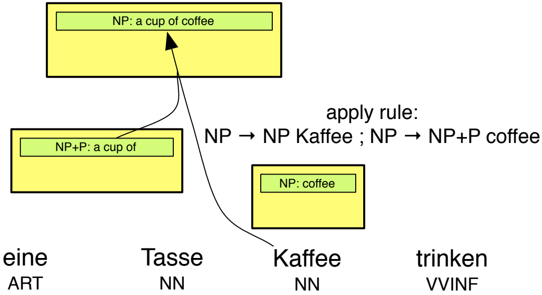
Chart Organization



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

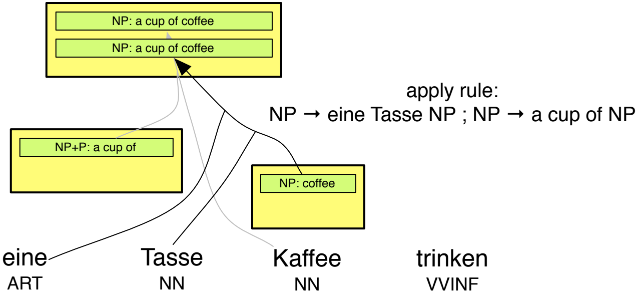
Dynamic Programming

Applying rule creates new hypothesis



Dynamic Programming

Another hypothesis

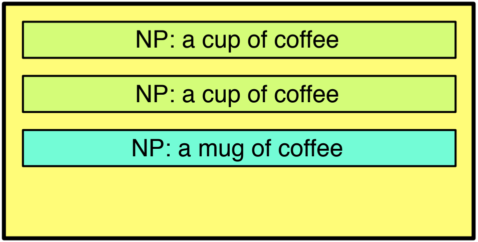


Both hypotheses are indistiguishable in future search

→ can be recombined

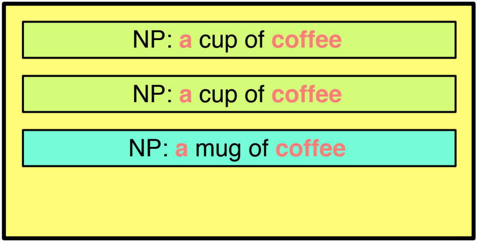
Recombinable States

Recombinable?



Recombinable States

Recombinable?



Yes, if max. 2-gram language model is used

Recombinability

Hypotheses have to match in

- span of input words covered
- output constituent label
- first n–1 output words

  not properly scored, since they lack context

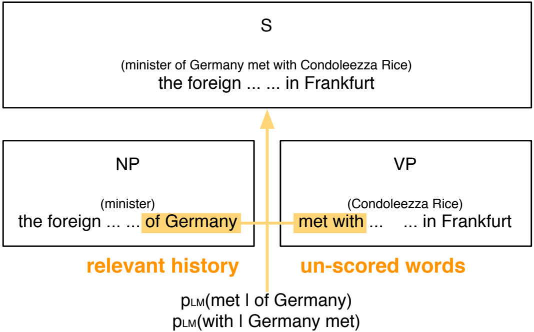
- last n–1 output words

  still affect scoring of subsequently added words,
  just like in phrase-based decoding

(n is the order of the n-gram language model)

Language Model Contexts

When merging hypotheses, internal language model contexts are absorbed



Stack Pruning

- Number of hypotheses in each chart cell explodes

$\Rightarrow$ 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
    $\rightarrow$ 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*

⇒ Unworkable

Solution

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

Storing Rules

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

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

- 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

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

Rule Applicability Check

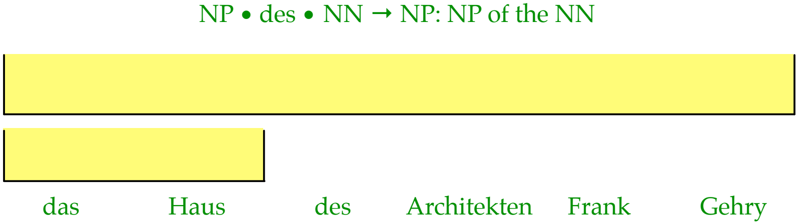
Trying to cover a span of six words with given rule

NP • des • NN → NP: NP of the NN

das Haus des Architekten Frank Gehry

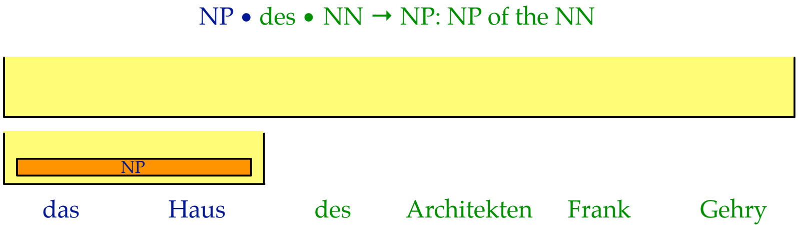
Rule Applicability Check

First: check for hypotheses with output constituent label NP



Rule Applicability Check

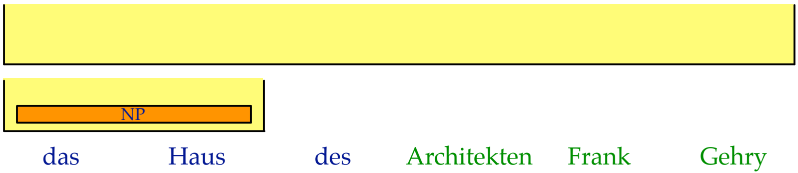
Found NP hypothesis in cell, matched first symbol of rule



Rule Applicability Check

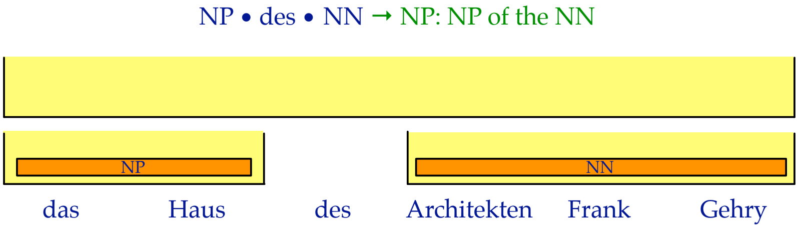
Matched word des, matched second symbol of rule

NP • des • NN → NP: NP of the NN



Rule Applicability Check

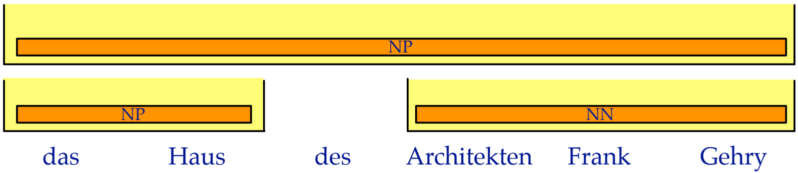
Found a NN hypothesis in cell, matched last symbol of rule



Rule Applicability Check

Matched entire rule → apply to create a NP hypothesis

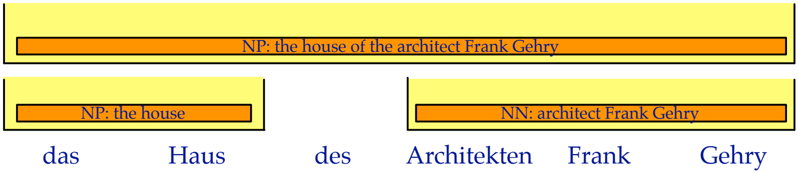
NP • des • NN → NP: NP of the NN



Rule Applicability Check

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

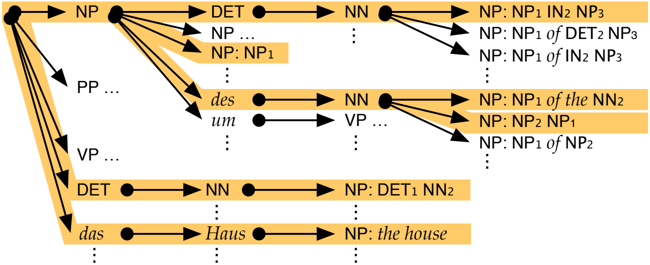
NP • des • NN → NP: NP of the NN



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

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

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

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

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

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

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

Dotted Rules: Key Insight

- If we can apply a rule like

$$\mathrm{p} \rightarrow \mathrm{A\ B\ C} \ \mid \ \mathrm{x}$$

  to a span

- Then we could have applied a rule like

$$\mathrm{q} \rightarrow \mathrm{A\ B} \ \mid \ \mathrm{y}$$

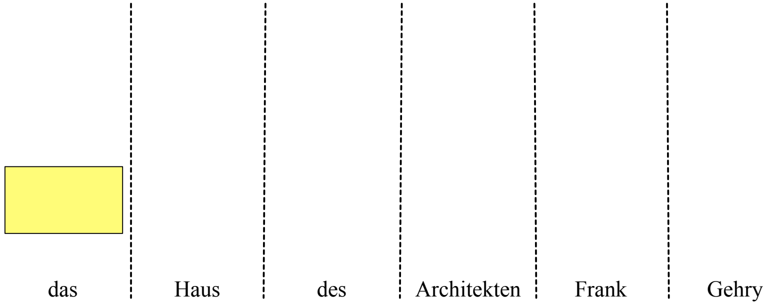
  to a sub-span with the same starting word

$\Rightarrow$ We can re-use rule lookup by storing $\mathrm{A\ B} \bullet$ (dotted rule)

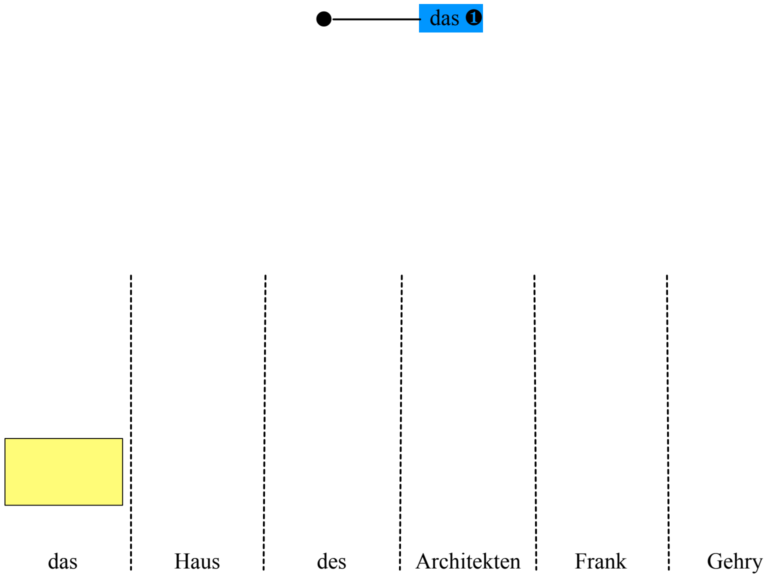
Finding Applicable Rules in Prefix Tree

das | Haus | des | Architekten | Frank | Gehry

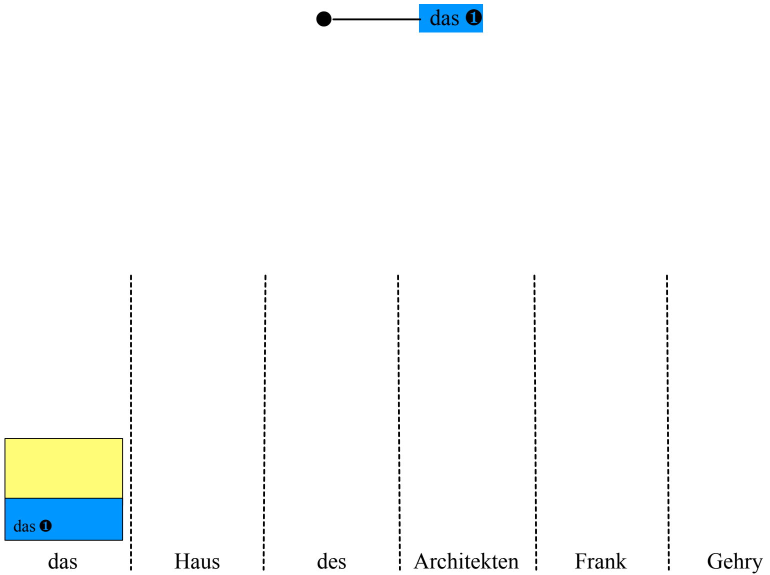
Covering the First Cell



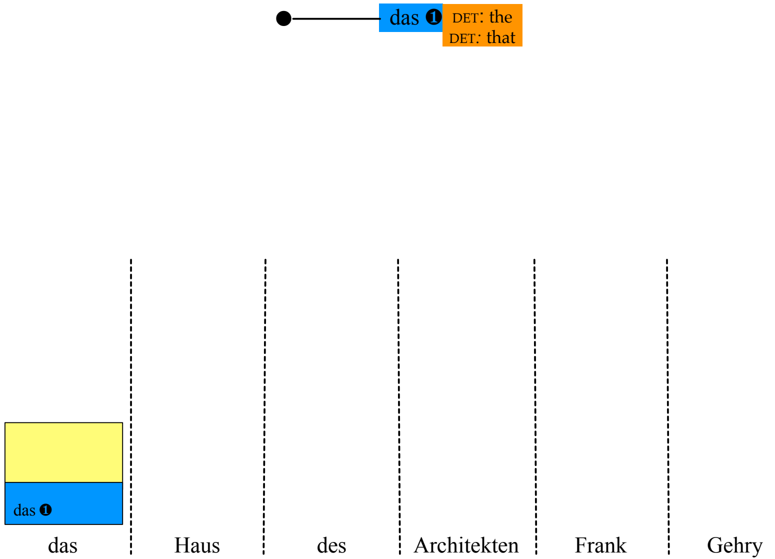
Looking up Rules in the Prefix Tree



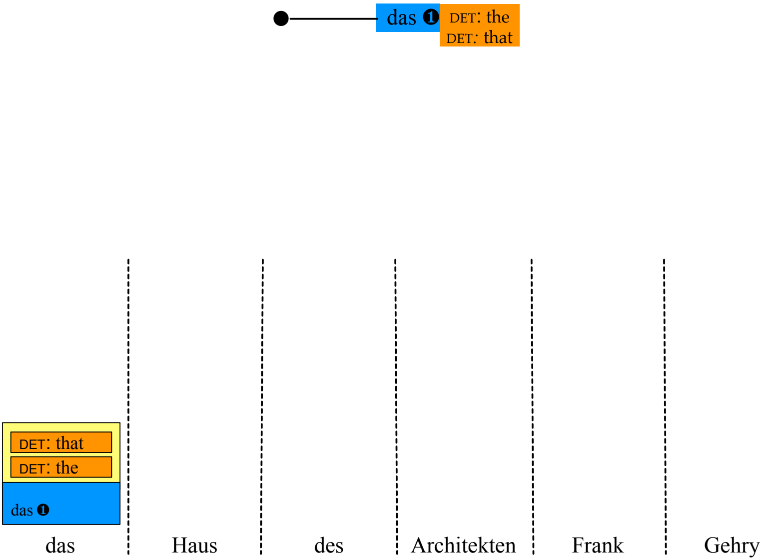
Taking Note of the Dotted Rule



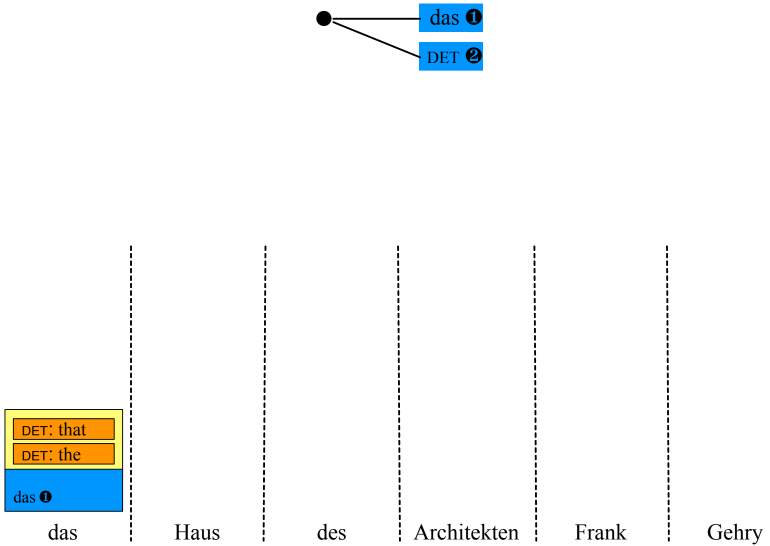
Checking if Dotted Rule has Translations



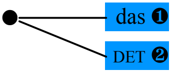
Applying the Translation Rules

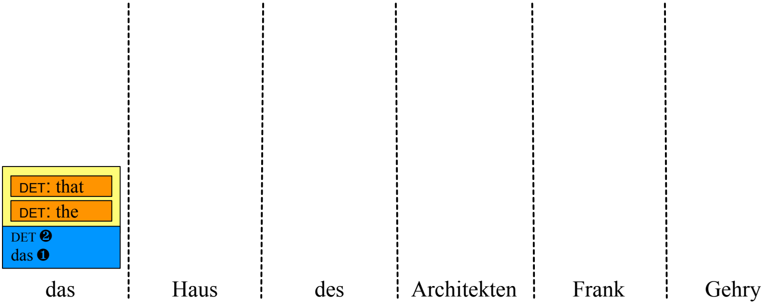


Looking up Constituent Label in Prefix Tree

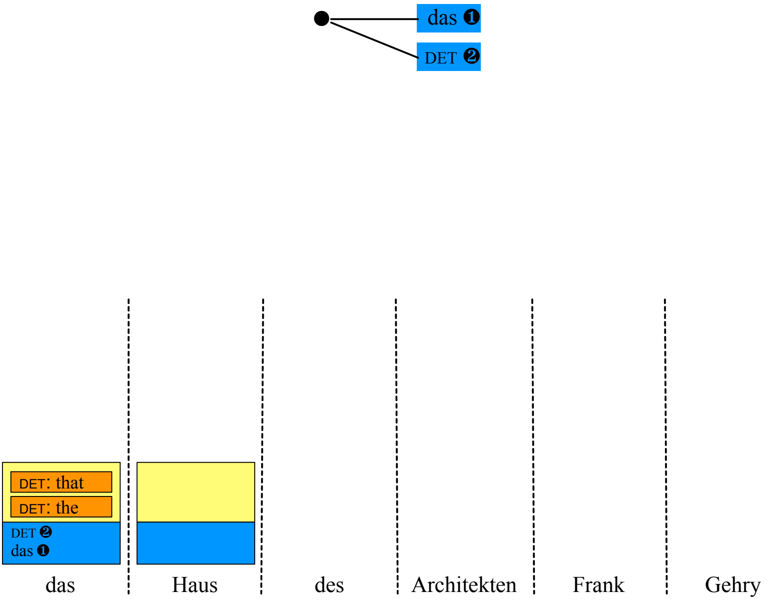


Add to Span's List of Dotted Rules

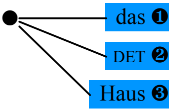


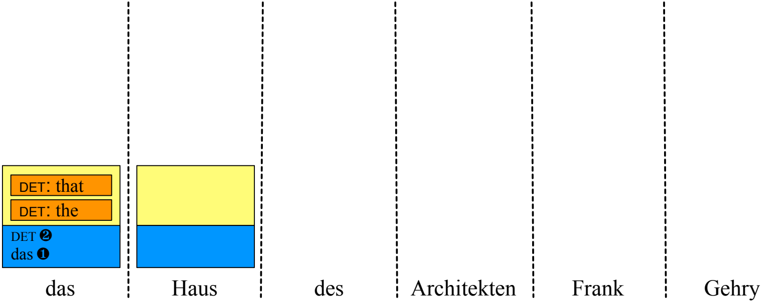


Moving on to the Next Cell

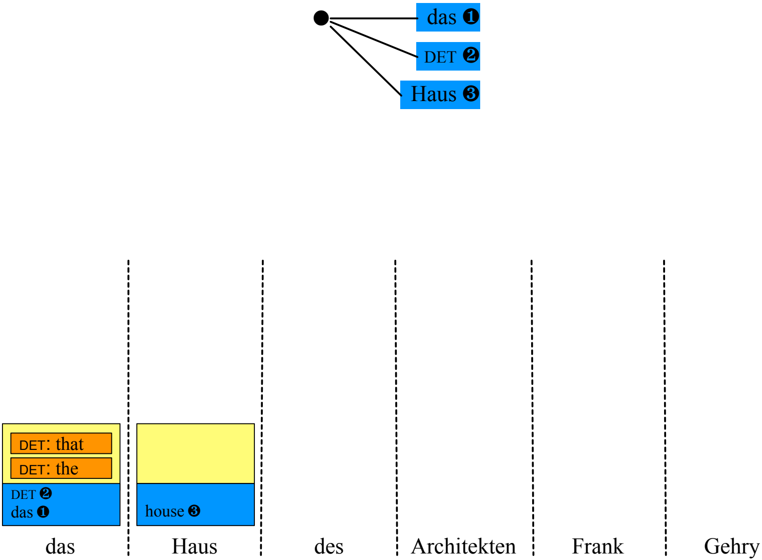


Looking up Rules in the Prefix Tree

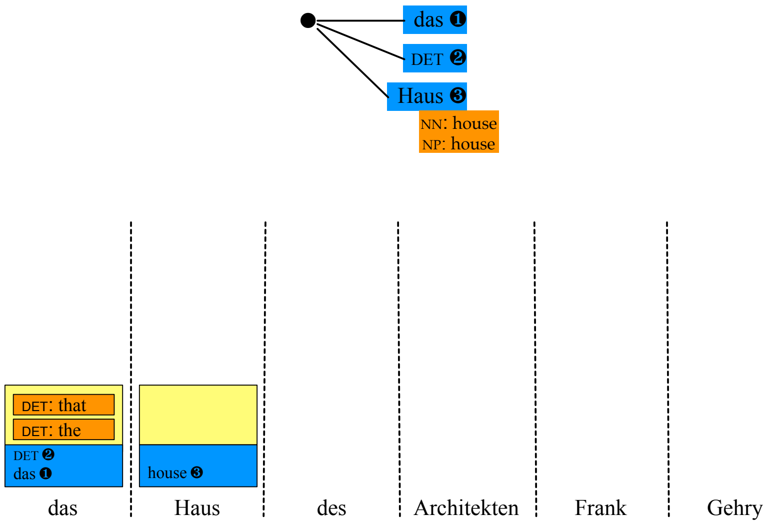




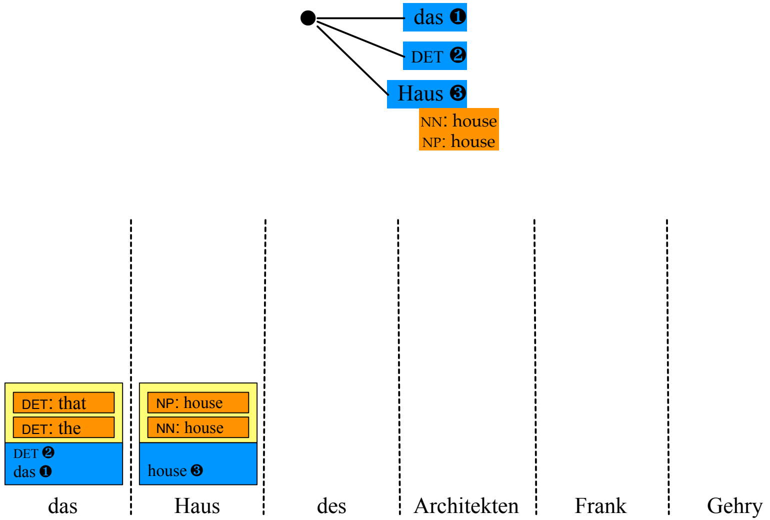
Taking Note of the Dotted Rule



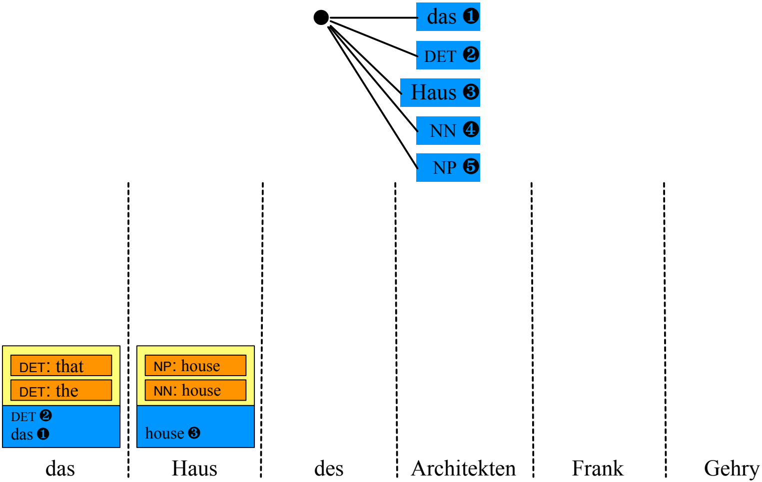
Checking if Dotted Rule has Translations



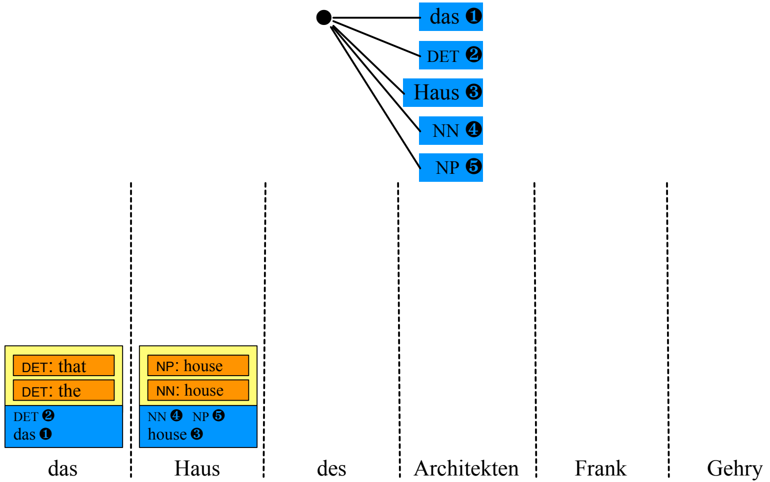
Applying the Translation Rules



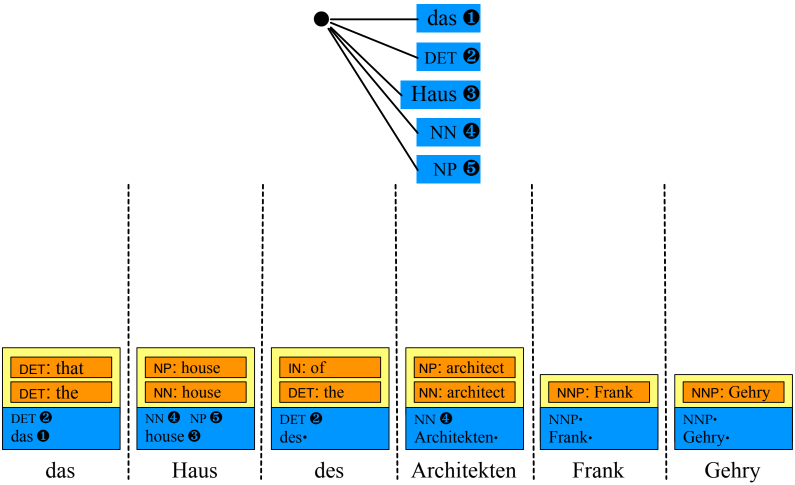
Looking up Constituent Label in Prefix Tree



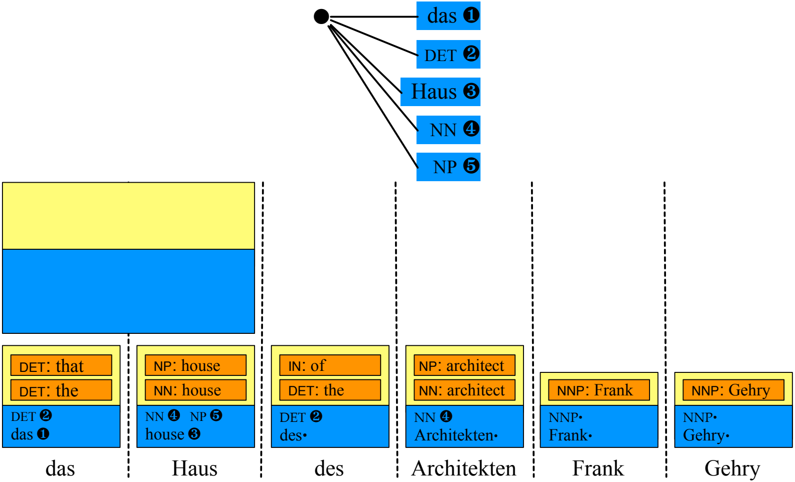
Add to Span's List of Dotted Rules



More of the Same



Moving on to the Next Cell

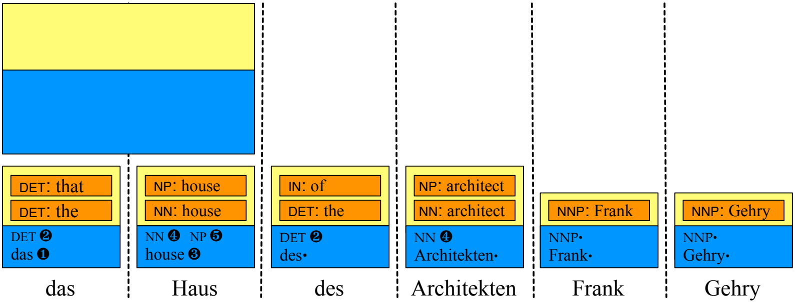


Covering a Longer Span

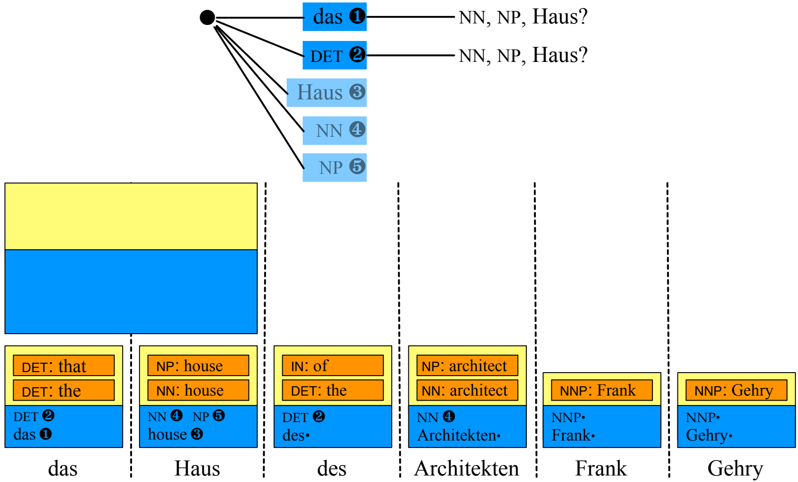
Cannot consume multiple words at once

All rules are extensions of existing dotted rules

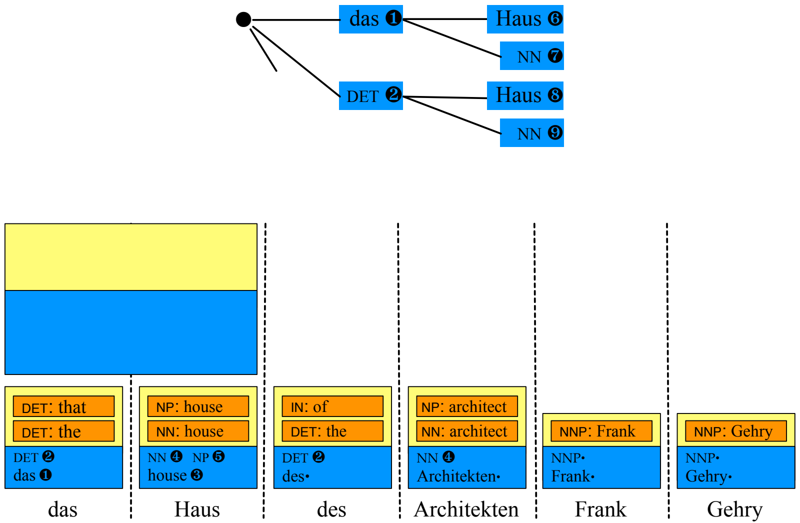
Here: only extensions of span over das possible



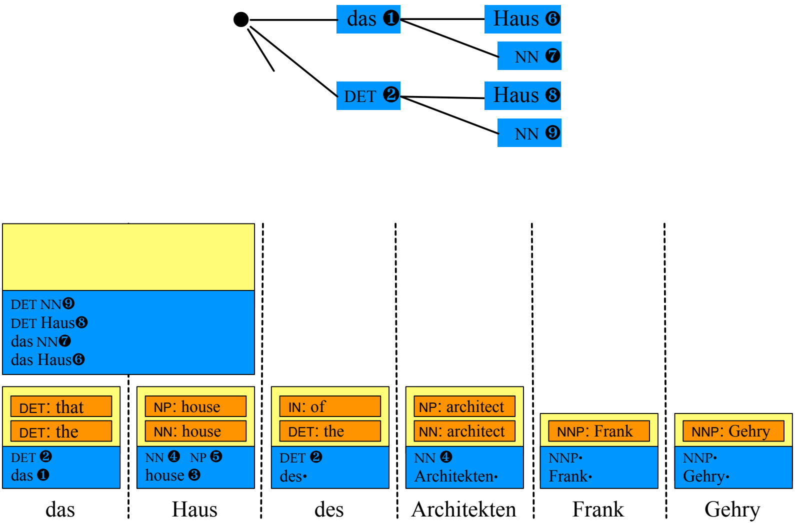
Extensions of Span over das



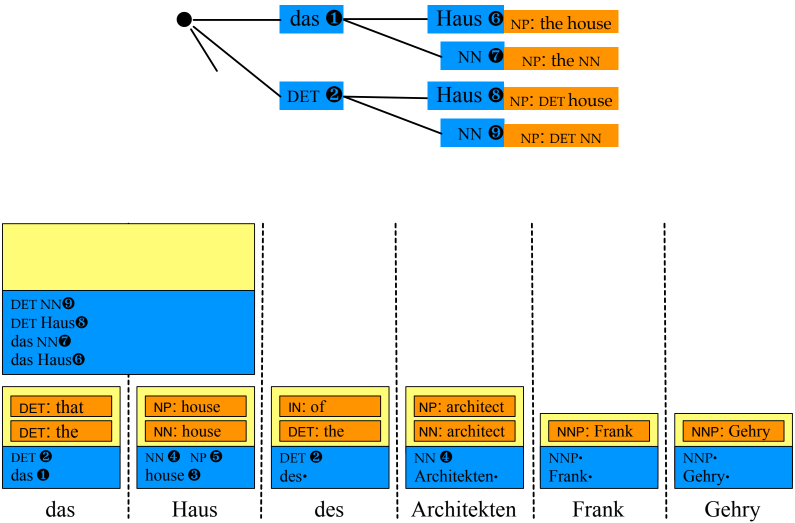
Looking up Rules in the Prefix Tree



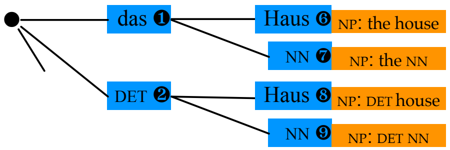
Taking Note of the Dotted Rule

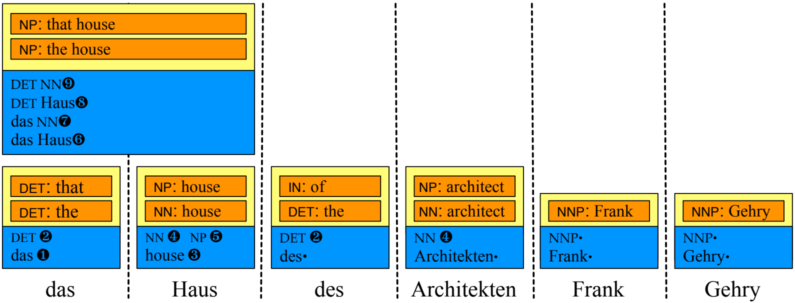


Checking if Dotted Rules have Translations

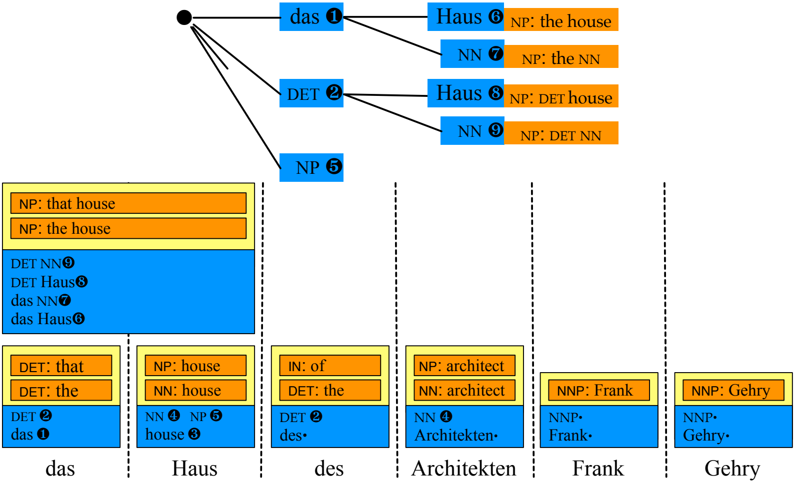


Applying the Translation Rules

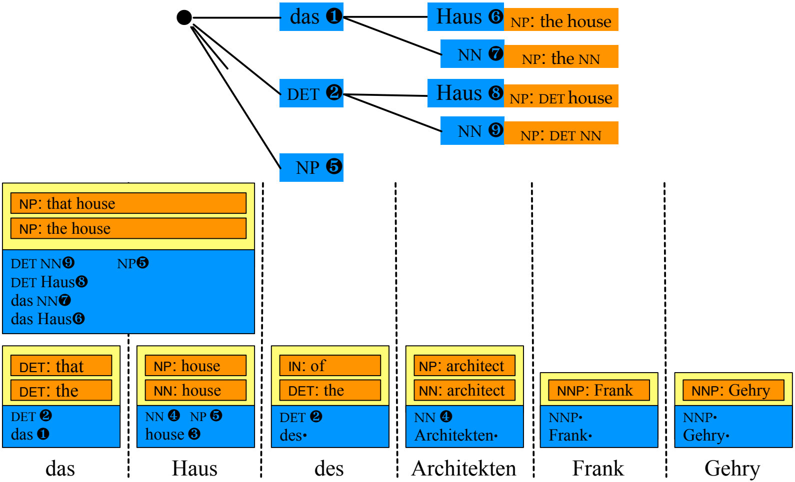




Looking up Constituent Label in Prefix Tree



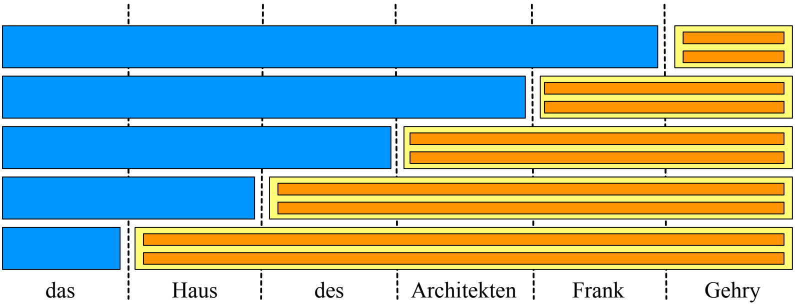
Add to Span's List of Dotted Rules



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

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

  - 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

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

  - will get out of hand even for syntax models

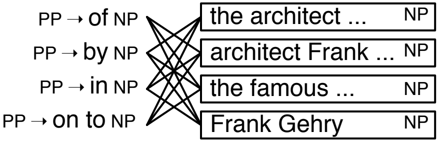
Where are we now?

- We know which rules apply
- We know where they apply (each non-terminal tied to a span)
- But there are still many choices
  - many possible translations
  - each non-terminal may match multiple hypotheses

  $\rightarrow$ number choices exponential with number of non-terminals

Rules with One Non-Terminal

Found applicable rules $\text{PP} \rightarrow \text{des X} \mid \ldots \text{ NP } \ldots$



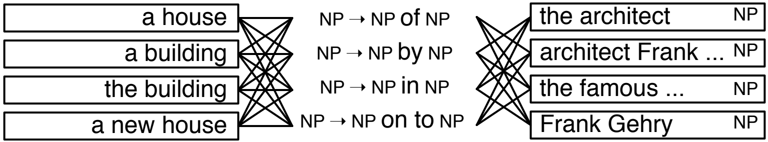
- Non-terminal will be filled any of h underlying matching hypotheses
- Choice of t lexical translations

$\Rightarrow$ Complexity $O(ht)$

(note: we may not group rules by target constituent label,
so a rule $\text{NP} \rightarrow \text{des X} \mid \text{the NP}$ would also be considered here as well)

Rules with Two Non-Terminals

Found applicable rule $\text{NP} \rightarrow \text{X}_1 \text{ des } \text{X}_2 \mid \text{NP}_1 \ldots \text{NP}_2$

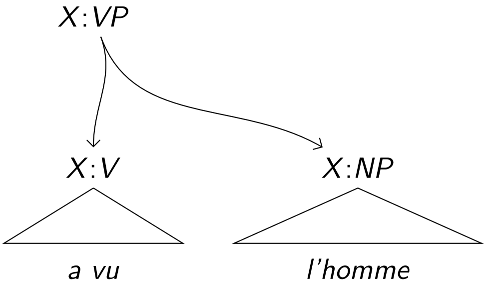


- Two non-terminal will be filled any of h underlying matching hypotheses each
- Choice of t lexical translations

$\Rightarrow$ Complexity $O(h^2 t)$ — a three-dimensional "cube" of choices

(note: rules may also reorder differently)

Filling a Constituent



Hyp	Score
seen	−3.8
saw	−4.0
view	−4.0

Hyp	Score
man	−3.6
the man	−4.3
some men	−6.3

Beam Search

	man -3.6	**the man -4.3**	**some men -6.3**
seen -3.8	seen man -8.8	seen the man -7.6	seen some men -9.5
saw -4.0	saw man -8.3	saw the man -6.9	saw some men -8.5
view -4.0	view man -8.5	view the man -8.9	view some men -10.8

Cube Pruning [Chiang, 2007]

	man -3.6	**the man -4.3**	**some men -6.3**
seen -3.8	Queue		
saw -4.0			
view -4.0			

Queue

Hypothesis	**Sum**
➔seen man	-3.8-3.6=-7.4

Cube Pruning [Chiang, 2007]

	man -3.6	**the man** -4.3	**some men** -6.3
seen -3.8	seen man -8.8	Queue	
saw -4.0	Queue		
view -4.0			

Queue

Hypothesis	**Sum**
➔saw man	-4.0-3.6=-7.6
seen the man	-3.8-4.3=-8.1

Cube Pruning [Chiang, 2007]

	man	**-3.6**	**the man -4.3**	**some men -6.3**
seen -3.8	seen man	-8.8	Queue	
saw -4.0	saw man	-8.3	Queue	
view -4.0	Queue			

Queue

	Hypothesis	**Sum**
➔	view man	-4.0-3.6=-7.6
	seen the man	-3.8-4.3=-8.1
	saw the man	-4.0-4.3=-8.3

Cube Pruning versus Beam Search

Same	Bottom-up with fixed-size beams
Different	Beam filling algorithm

Queue of Cubes

- Several groups of rules will apply to a given span
- Each of them will have a cube
- We can create a queue of cubes

⇒ Always pop off the most promising hypothesis, regardless of cube

- May have separate queues for different target constituent labels

Bottom-Up Chart Decoding Algorithm

```
for all spans (bottom up) do
    extend dotted rules
    for all dotted rules do
        find group of applicable rules
        create a cube for it
        create first hypothesis in cube
        place cube in queue
    end for
    for specified number of pops do
        pop off best hypothesis of any cube in queue
        add it to the chart cell
        create its neighbors
    end for
    extend dotted rules over constituent labels
end for
```

Two-Stage Decoding

- First stage: decoding without a language model (-LM decoding)
  - may be done exhaustively
  - eliminate dead ends
  - optionably prune out low scoring hypotheses
- Second stage: add language model
  - limited to packed chart obtained in first stage
- Note: essentially, we do two-stage decoding for each span at a time

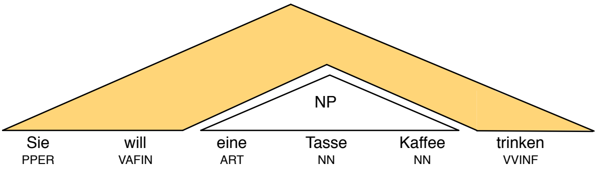
Coarse-to-Fine

- Decode with increasingly complex model

- Examples
  - reduced language model [Zhang and Gildea, 2008]
  - reduced set of non-terminals [DeNero et al., 2009]
  - language model on clustered word classes [Petrov et al., 2008]

Outside Cost Estimation

- Which spans should be more emphasized in search?
- Initial decoding stage can provide outside cost estimates



- Use min/max language model costs to obtain admissible heuristic
  (or at least something that will guide search better)

Open Questions

- Where does the best translation fall out the beam?
- Are particular types of rules too quickly discarded?
- Are there systemic problems with cube pruning?

Summary

- Synchronous context free grammars
- Extracting rules from a syntactically parsed parallel corpus
- Bottom-up decoding
- Chart organization: dynamic programming, stacks, pruning
- Prefix tree for rules
- Dotted rules
- Cube pruning