Hierarchical Models and Chart-Based Decoding

Barry Haddow

(Based on slides by Philipp Koehn and Kenneth Heafield)

12 September, 2013

Tree-Based Models

- The models we've seen so far operate on sequences of words
- Many translation problems can be best explained by pointing to syntax
  - reordering, e.g., verb movement in German–English translation
  - long distance agreement (e.g., subject-verb) in output

⇒ Translation models based on tree representation of language
  - significant ongoing research
  - state-of-the art for some language pairs

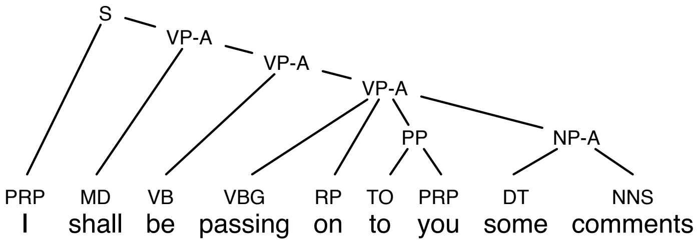
Phrase Structure Grammar

- Phrase structure
  - noun phrases: the big man, a house, ...
  - prepositional phrases: at 5 o'clock, in Edinburgh, ...
  - verb phrases: going out of business, eat chicken, ...
  - adjective phrases, ...
- Context-free Grammars (CFG)

  non-terminals: phrase structure labels, part-of-speech tags
  terminals: words
  rules: rewrite non-terminal as sequence of Ts and NT

  e.g. $\text{NP} \rightarrow \text{DET NN}$
- Probabilistic Context-free Grammars (PCFG)
  - Attach probabilities to rules

Parse Tree



Phrase structure grammar tree for an English sentence
(as produced Collins' parser)

Synchronous Context Free Grammar

- English rule

$$\text{NP} \rightarrow \text{DET JJ NN}$$

- French rule

$$\text{NP} \rightarrow \text{DET NN JJ}$$

- Synchronous rule (indices indicate alignment):

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

Synchronous Grammar Rules

- Nonterminal rules

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

- Terminal rules

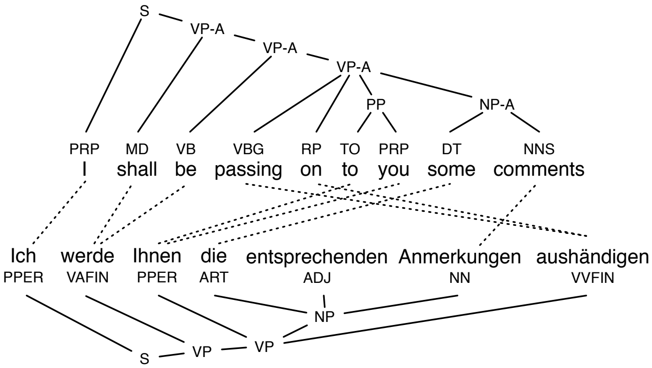
$$\text{N} \rightarrow \text{maison} \mid \text{house}$$

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

- Mixed rules

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

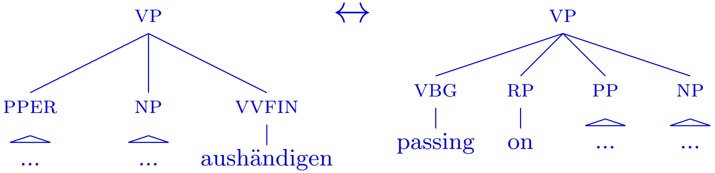
Aligned Tree Pair



Phrase structure grammar trees with word alignment
(German–English sentence pair.)

Reordering Rule

- Subtree alignment



- Synchronous grammar rule

$$\text{VP} \rightarrow \text{PPER}_1 \ \text{NP}_2 \ \text{aushändigen} \ \mid \ \text{passing on PP}_1 \ \text{NP}_2$$

- Note:
  - one word aushändigen mapped to two words passing on ok
  - but: fully non-terminal rule not possible

Rules with Internal Structure

- Subtree alignment

- Synchronous grammar rule (stripping out English internal structure)

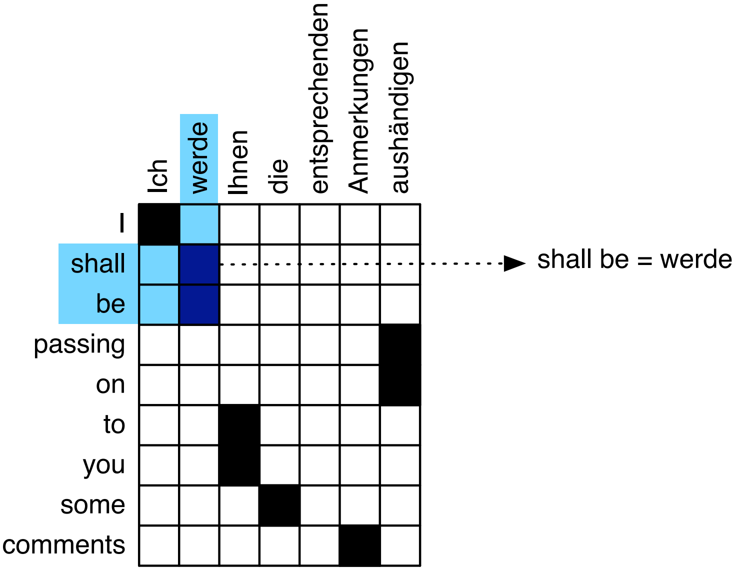
$$\text{PRO/PP} \rightarrow \text{Ihnen} \mid \text{to you}$$

- Rule with internal structure (Synchronous Tree Substitution Grammar)

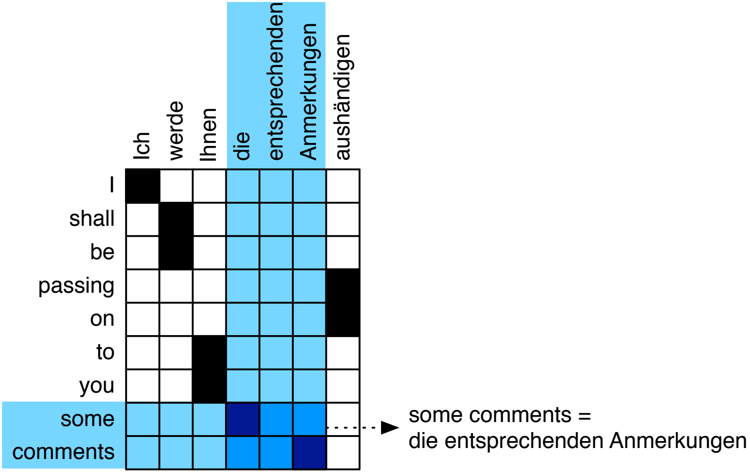
Learning Synchronous Grammars

- Extract rules from a word-aligned parallel corpus
- Hierarchical phrase-based model (hiero)
  - only one non-terminal symbol X
  - no linguistic syntax, just a formally syntactic model
- Synchronous phrase structure model
  - non-terminals for words and phrases: NP, VP, PP, ADJ, ...
  - corpus must also be parsed with syntactic parser
  - restrict extraction to rules compatible with parse
  - string-to-tree, tree-to-string, tree-to-tree, . . .

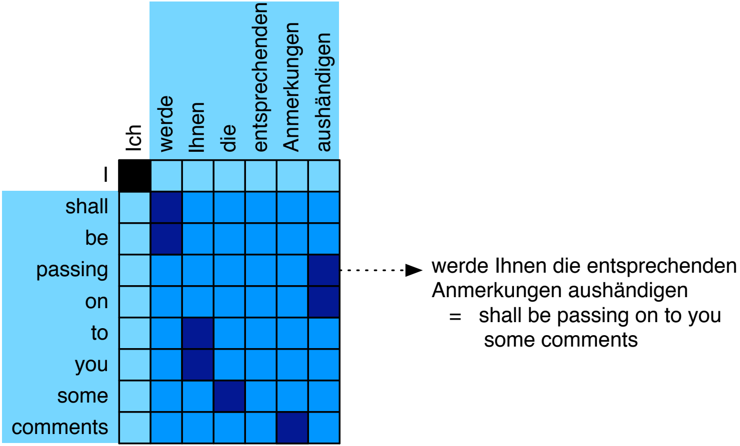
Extracting Phrase Translation Rules



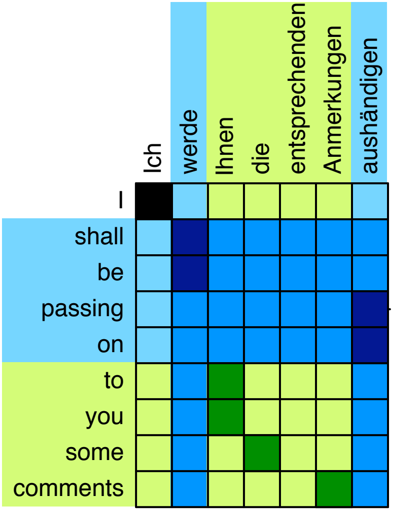
Extracting Phrase Translation Rules



Extracting Phrase Translation Rules



Extracting Hierarchical Phrase Translation Rules



subtracting subphrase

werde X aushändigen
= shall be passing on X

Hierarchical Rule extraction

- All phrase-pairs licensed by PBMT heuristics
- Recursively add *hierarchical* rules
  - So if we have:

$$\mathrm{X} \rightarrow \mathrm{abc} \mid \mathrm{pqrs} \qquad \mathrm{X} \rightarrow \mathrm{b} \mid \mathrm{qr}$$

  - We can add:

$$\mathrm{X} \rightarrow \mathrm{aXc} \mid \mathrm{pXs}$$

- Continue until no more rules can be added
- Rule probabilities derived from frequencies

- Syntax-based models require non-terminals to be constituents

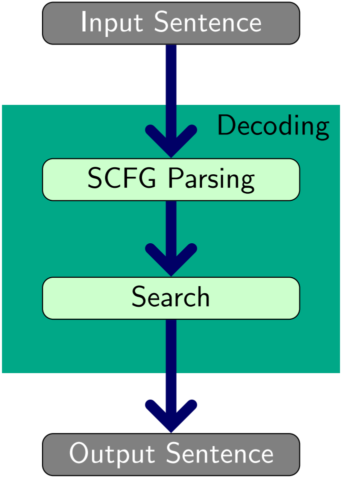
Hiero Extraction in Practice

- Removal of multiple sub-phrases leads to rules with multiple non-terminals, such as:

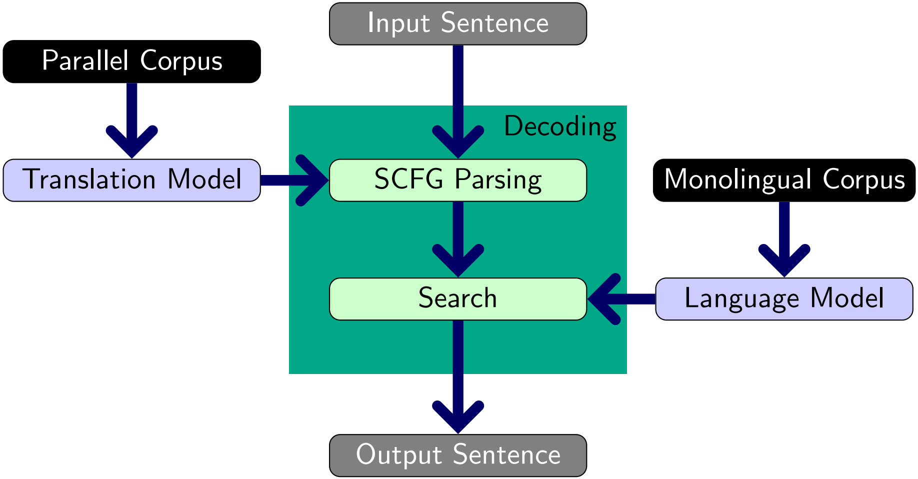
$$\text{Y} \rightarrow \text{X}_1 \ \text{X}_2 \ \mid \ \text{X}_2 \ \textit{of} \ \text{X}_1$$

- Typical restrictions to limit complexity
  - at most 2 nonterminal symbols
  - at least 1 but at most 5 words per language
  - span at most 15 words (counting gaps)
- Size of europarl-derived fr-en rule table:
  - PB: 100M Hiero: 800M

Overview of Syntactic Decoding



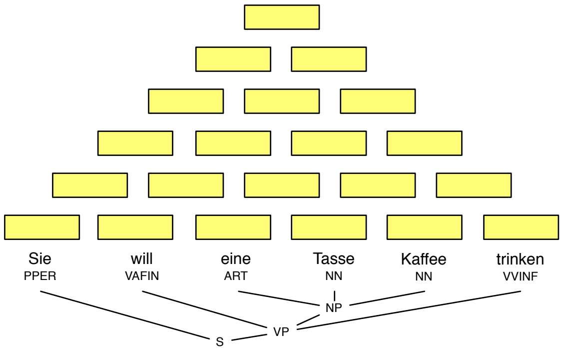
Overview of Syntactic Decoding



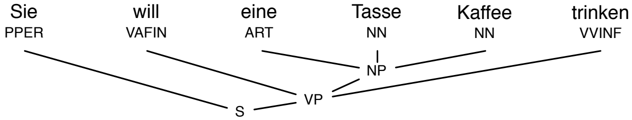
Syntactic Decoding

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

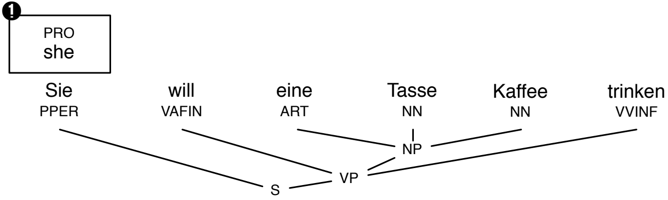


Syntax Decoding



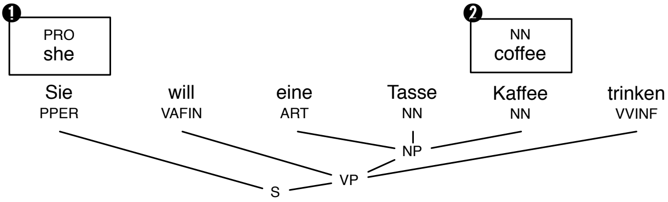
German input sentence with tree

Syntax Decoding



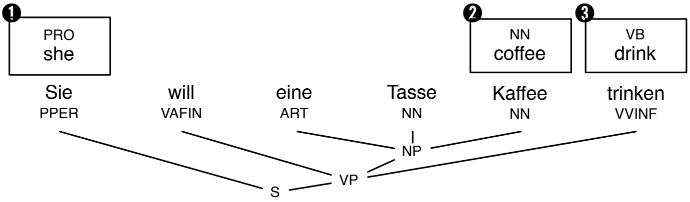
Purely lexical rule: filling a span with a translation (a constituent)

Syntax Decoding



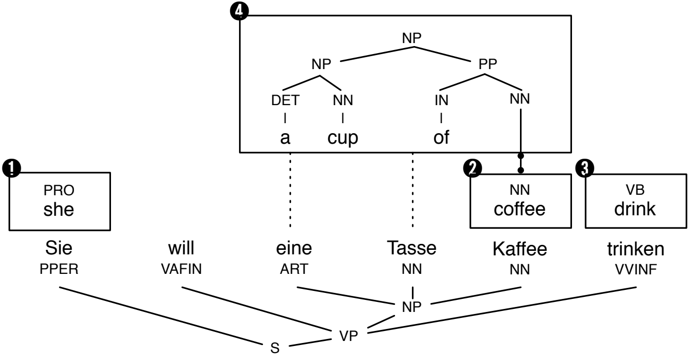
Purely lexical rule: filling a span with a translation (a constituent)

Syntax Decoding



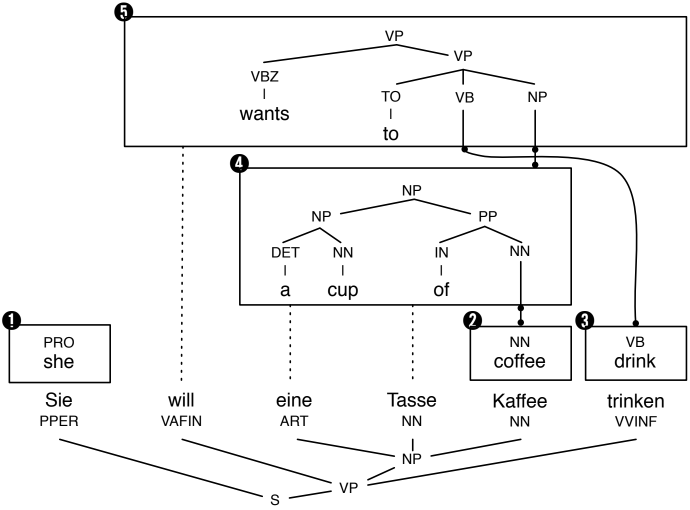
Purely lexical rule: filling a span with a translation (a constituent)

Syntax Decoding



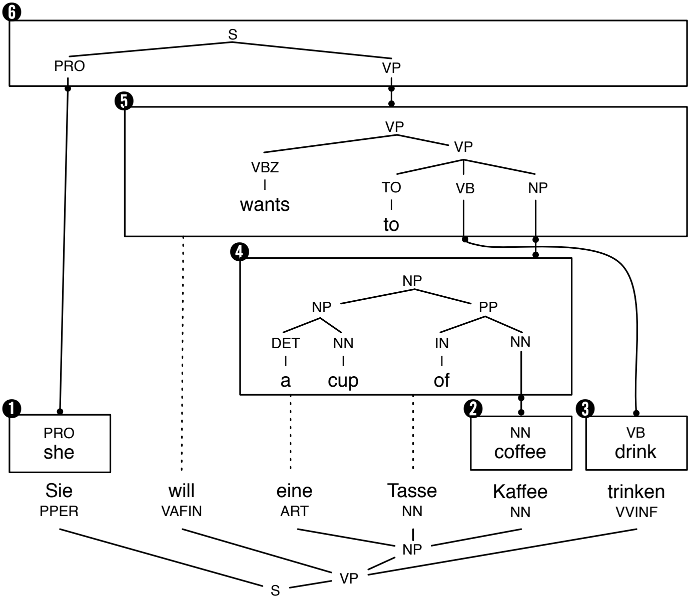
Complex rule: matching underlying constituent spans, and covering words

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

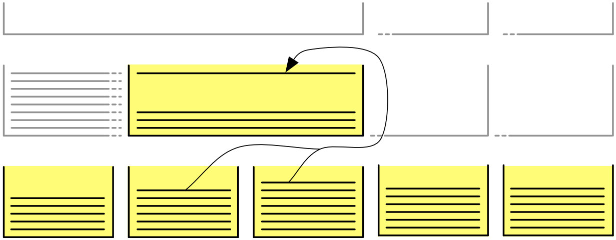
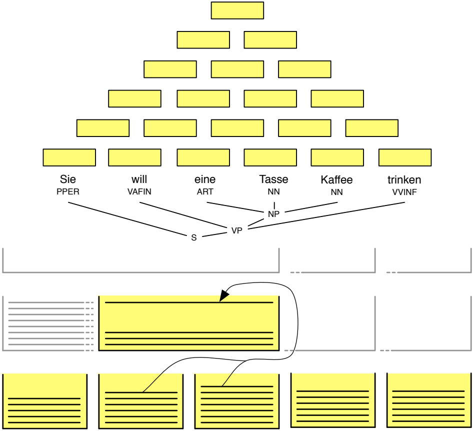


Chart Organization



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

Naive Algorithm

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

```
for all spans [start,end] (bottom up) do
    for all sequences s of hypotheses and words in span [start,end] do
        for all rules r do
            if rule r applies to chart sequence s then
                create new hypothesis c
                add hypothesis c to chart
            end if
        end for
    end for
end for
return English translation e from best hypothesis in span [0,l_f]
```

Naive Algorithm

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

for all spans [start,end] (bottom up) **do**
    for all sequences *s* of hypotheses and words in span [start,end] **do**
        for all rules *r* **do**
            if rule *r* applies to chart sequence *s* **then**
                create new hypothesis *c*
                add hypothesis *c* to chart
            end if
        end for
    end for
end for
return English translation $\mathbf{e}$ from best hypothesis in span $[0, l_f]$

Many subspan sequences

Naive Algorithm

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

for all spans [start,end] (bottom up) **do**
    for all sequences *s* of hypotheses and words in span [start,end] **do**
        for all rules *r* **do**
            if rule *r* applies to chart sequence *s* **then**
                create new hypothesis *c*
                add hypothesis *c* to chart
            end if
        end for
    end for
end for
return English translation **e** from best hypothesis in span $[0, l_f]$

Many rules

Naive Algorithm

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

```
for all spans [start,end] (bottom up) do
  for all sequences s of hypotheses and words in span
  [start,end] do
    for all rules r do
      if rule r applies to chart sequence s then
        create new hypothesis c
        add hypothesis c to chart
      end if
    end for
  end for
end for
return English translation e from best hypothesis in span [0,l_f]
```

Checking rule application expensive

Naive Algorithm

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

for all spans [start,end] (bottom up) **do**
    for all sequences s of hypotheses and words in span [start,end] **do**
        for all rules r **do**
            if rule r applies to chart sequence s **then**
                create new hypothesis c
                add hypothesis c to chart
            end if
        end for
    end for
end for
return English translation $\mathbf{e}$ from best hypothesis in span $[0, l_f]$

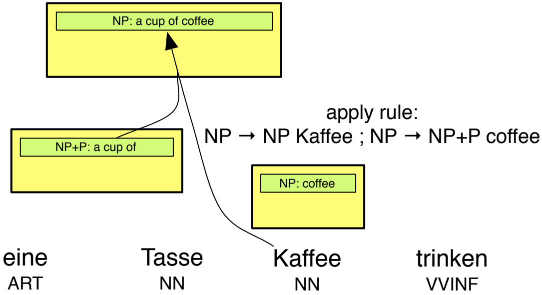
Scoring rules expensive $\rightarrow$ LM

Solutions

- Recombination
- Stack Pruning
- Prefix tree and Dotted Rules
- Cube pruning

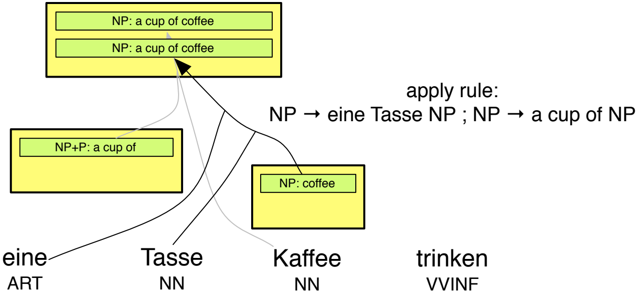
Dynamic Programming

Rule application creates new hypothesis



Dynamic Programming

Another hypothesis

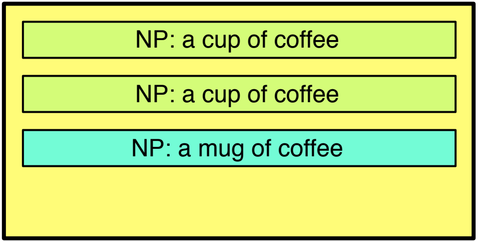


Both hypotheses are indistiguishable in future search

→ can be recombined

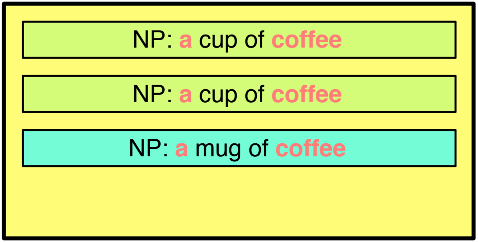
Recombinable States

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)

Stack Pruning

- Number of hypotheses in each chart cell explodes
  $\rightarrow$ Only keep a fixed number
- 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?
    (predict how useful constituent will be later on)

Storing Rules

- Need to quickly check which rules apply
  $\rightarrow$ match to available hypotheses and input words
- Example rule

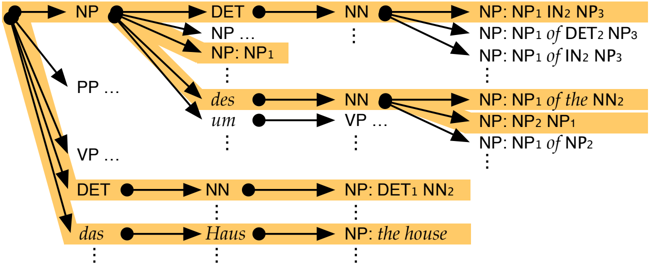
$$\text{NP} \rightarrow \text{X}_1 \text{ des } \text{X}_2 \ \mid \ \text{NP}_1 \text{ of the } \text{NN}_2$$

- Check for applicability
  - Subspan with constituent label NP?
  - Input word des?
  - Subspan NN?
- Does it apply? – check this sequence:

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

- Use Prefix Tree $\rightarrow$ can check many rules at once

Prefix Tree for Rules



Highlighted Rules

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

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

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

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

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

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

Optimising Lookups – Dotted Rules

- If we are trying to match a rule like

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

  ... then it helps if we already matched $A\ B$ to a subspan.

- So store partial matches of the prefix tree

- These are known as Dotted Rules

  $A\ B\ \bullet$

Where are we now?

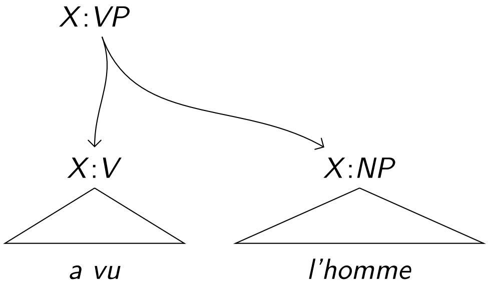
- Avoid creating hypotheses that cannot be optimal
  - Using recombination
- Only keep best scoring hypothesis in each cell
  - Stack pruning
- Efficiently organise rules for lookup
  - Prefix tree and dotted rules

Where are we now?

- Avoid creating hypotheses that cannot be optimal
  - Using recombination
- Only keep best scoring hypothesis in each cell
  - Stack pruning
- Efficiently organise rules for lookup
  - Prefix tree and dotted rules
- But LM lookup makes hypothesis combination so slow!

$\rightarrow$ $p(\text{saw}|\text{the man}) \neq p(\text{saw})p(\text{the}|\text{man})$

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

Naive 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

	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

	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

	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

	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	view man -8.5	Queue	

Queue

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

Cube Pruning

	man -3.6	**the man** -4.3	**some men** -6.3
seen -3.8	seen man -8.8	seen the man -7.6	Queue
saw -4.0	saw man -8.3	Queue	
view -4.0	view man -8.5	Queue	

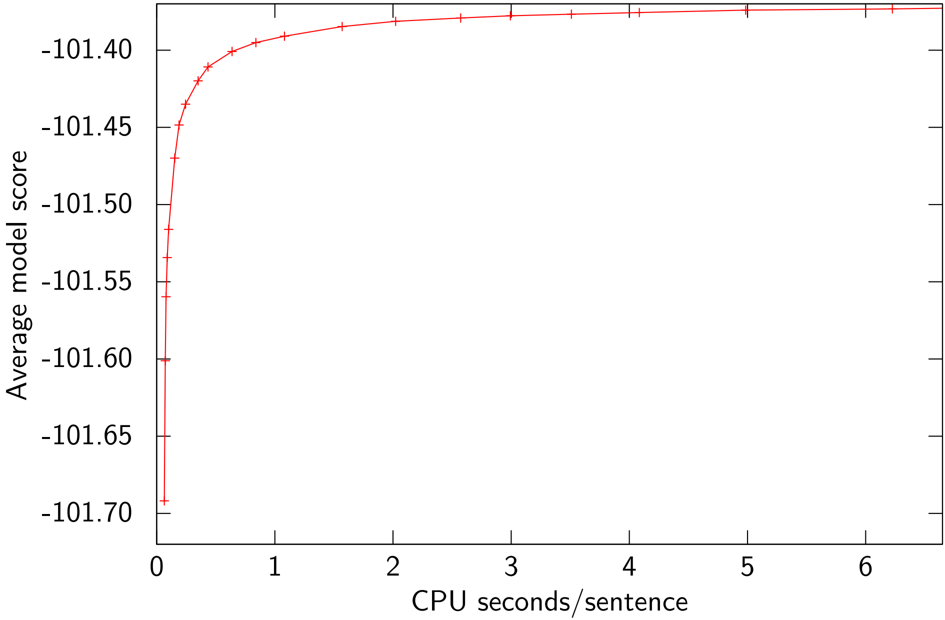
Queue

Hypothesis	**Sum**
→ saw the man	-4.0-4.3= -8.3
view the man	-4.0-4.3= -8.3
seen some men	-3.8-6.3=-10.1

Cube Pruning versus Beam Search

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

Cube Pruning: Speed vs. Accuracy



Many Cubes

- Could be several source-side matches for given span
- Create a cube for each one
- One queue per cube – or single queue
  $\rightarrow$ Always pop most promising hypothesis

One Stage or Two Stage Decoding

- First stage: decoding without a language model (-LM decoding)
  - Can 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
- Can do a single pass (interleaved)

Vs.

cdec does 2 passes

but Moses does 1!

Summary

- Synchronous context free grammars
- Rule extraction from aligned corpus
- Bottom-up decoding
- Chart organization: dynamic programming, stacks, pruning
- Prefix tree for rules
- Dotted rules
- Cube pruning