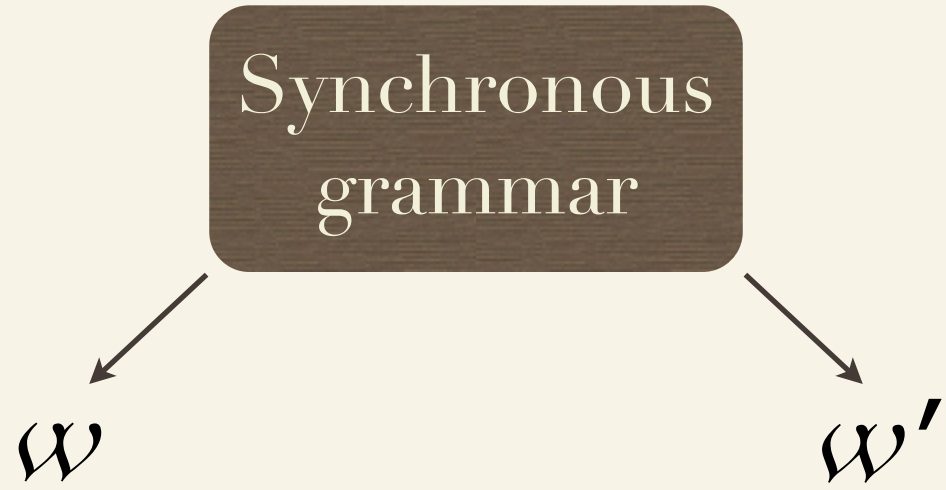
Synchronous Grammars

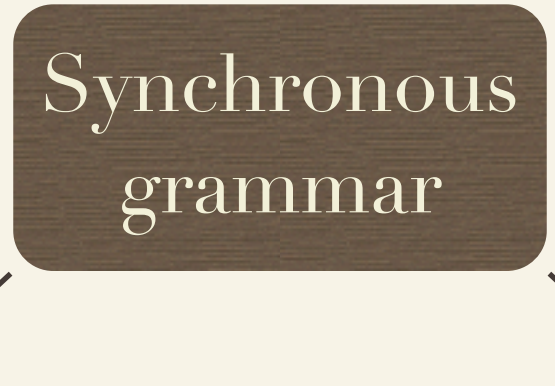
Synchronous grammars

are a way of simultaneously generating pairs of recursively related strings



Synchronous grammars

were originally invented for
programming language compilation

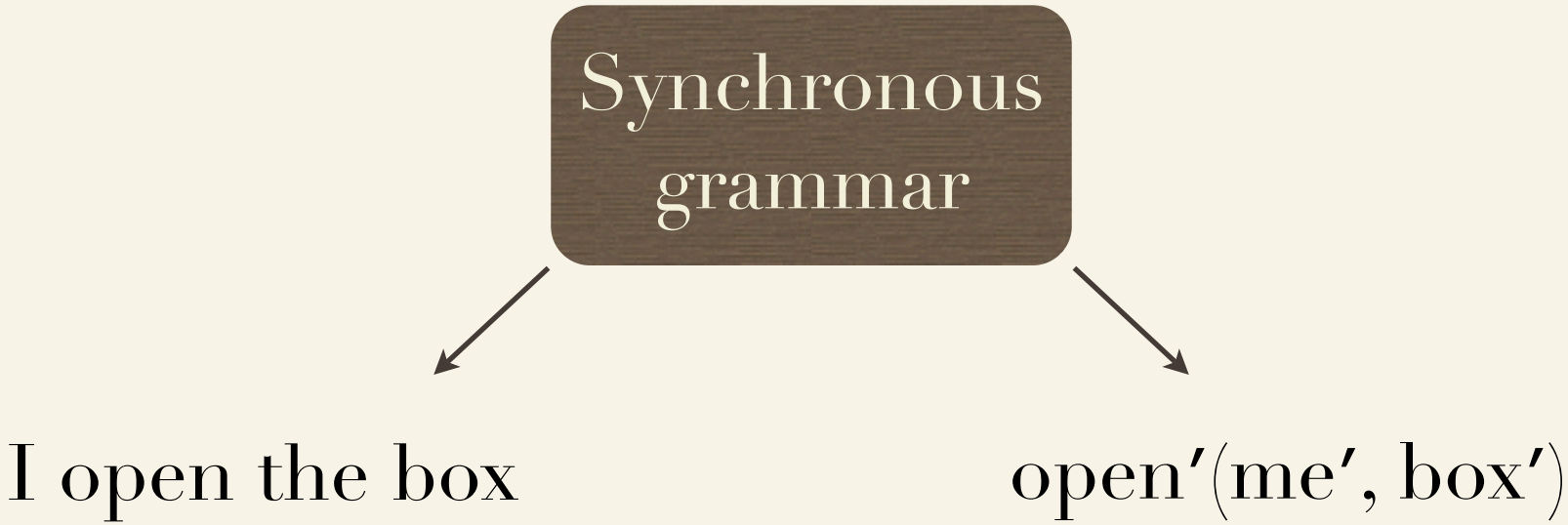



```
for i := 1 to 10 do
begin
    n := n + i
end
```

```
mov ax, 1
loop: add bx, ax
cmp ax, 10
jle loop
```

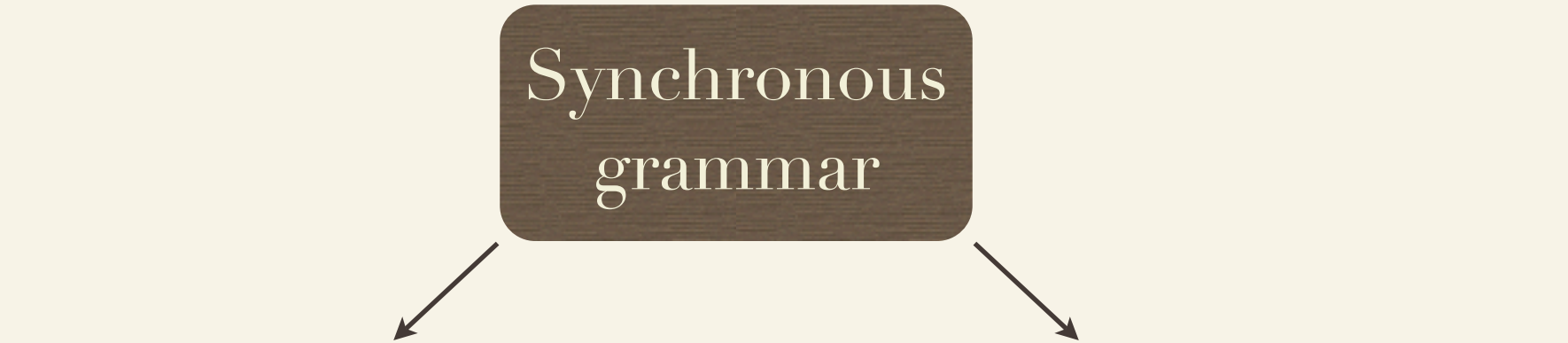
Synchronous grammars

have been proposed as a way of doing semantic interpretation



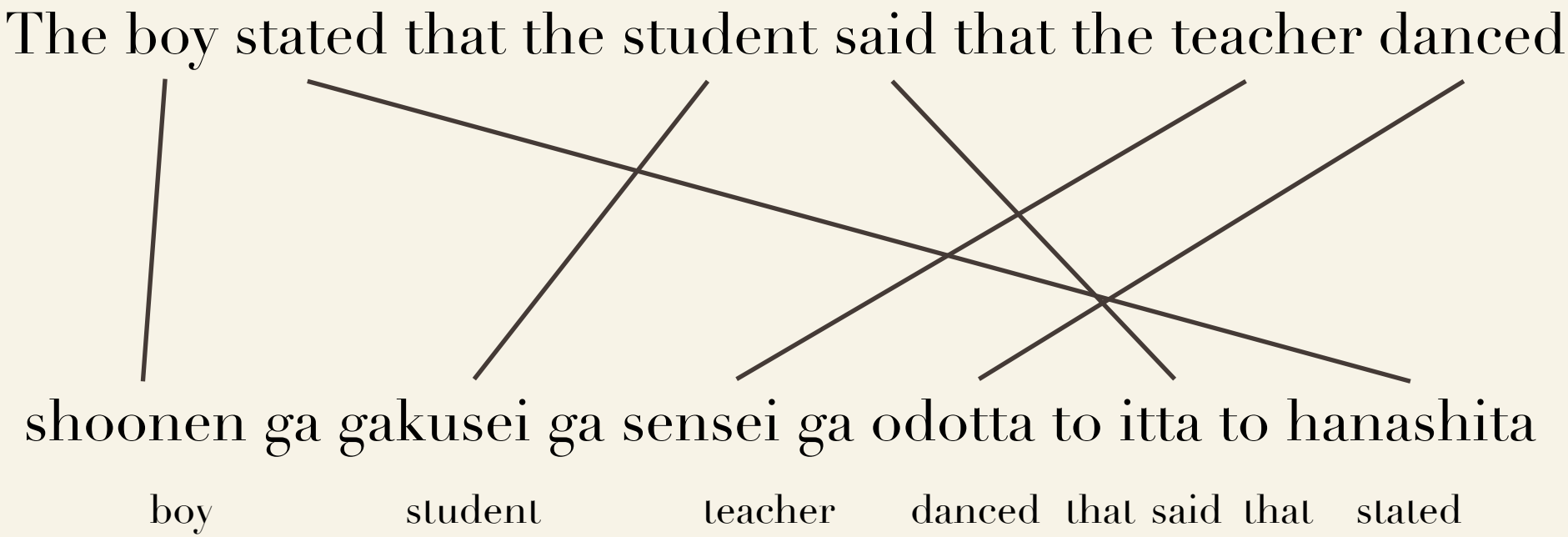
Synchronous grammars

have been used for syntax-based machine translation



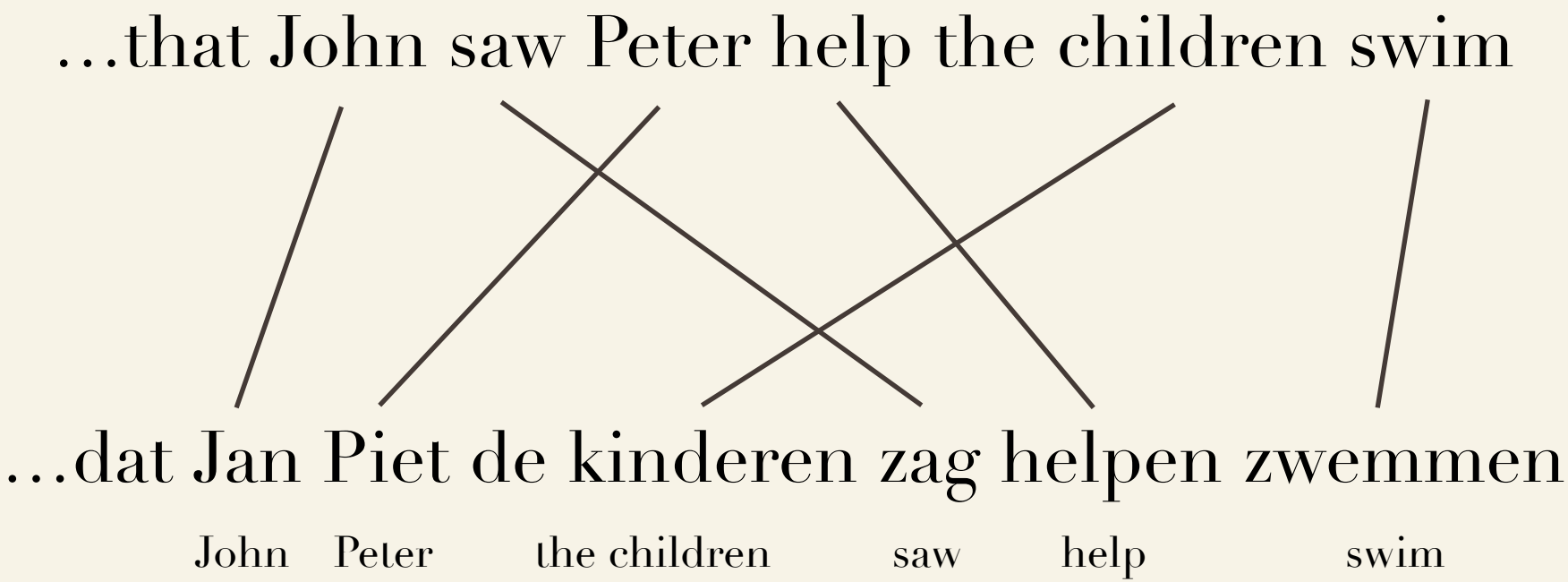
Synchronous grammars

can do much fancier transformations than finite-state methods



Synchronous grammars

can do much fancier transformations
than finite-state methods



Overview

- Definitions
- Properties
- Algorithms
- Extensions

Definitions

Context-free grammars

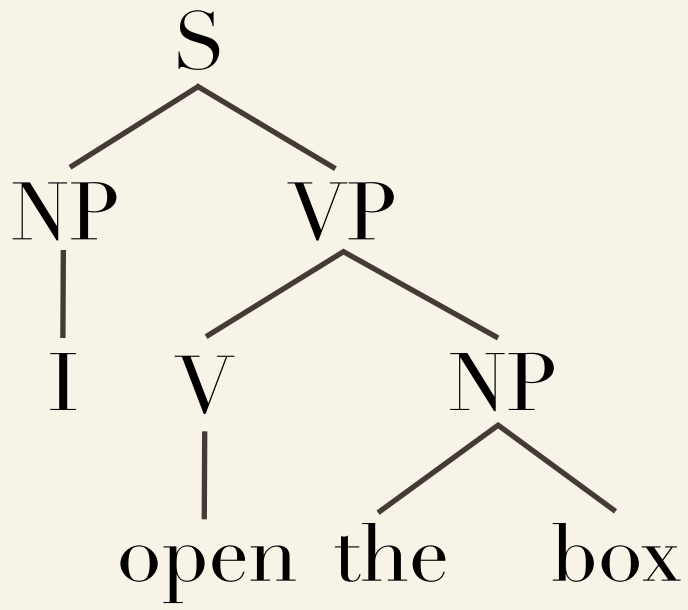
S → NP VP

NP → I

NP → the box

VP → V NP

V → open



Context-free grammars

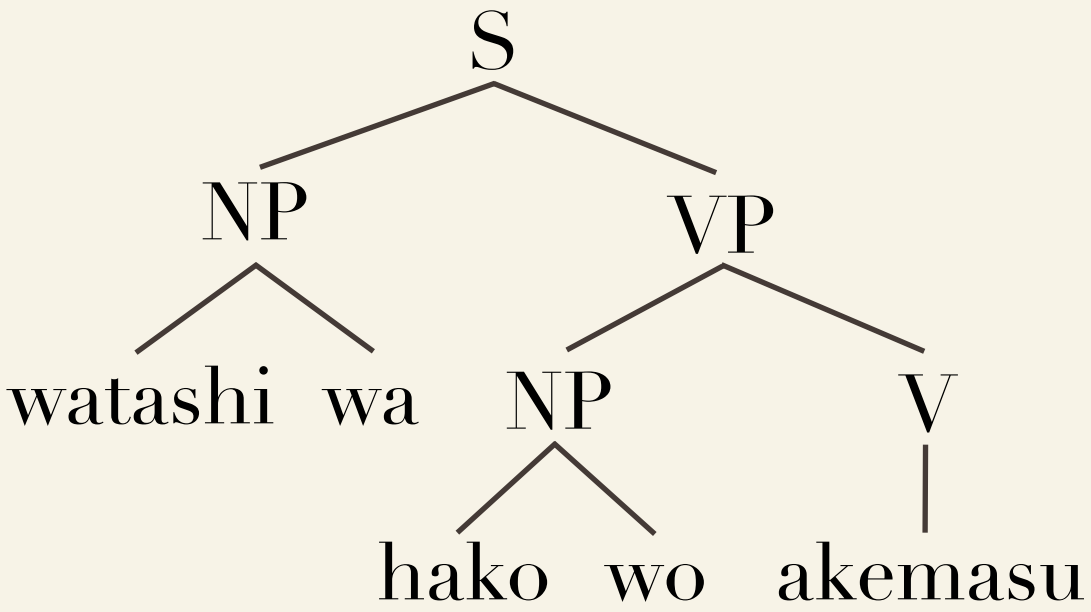
S → NP VP

NP → watashi wa

NP → hako wo

VP → NP V

V → akemasu



Synchronous CFGs

S → NP VP	S → NP VP
NP → I	NP → watashi wa
NP → the box	NP → hako wo
VP → V NP	VP → NP V
V → open	V → akemasu

Synchronous CFGs

$$S \rightarrow NP_1\ VP_2,\ NP_1\ VP_2$$

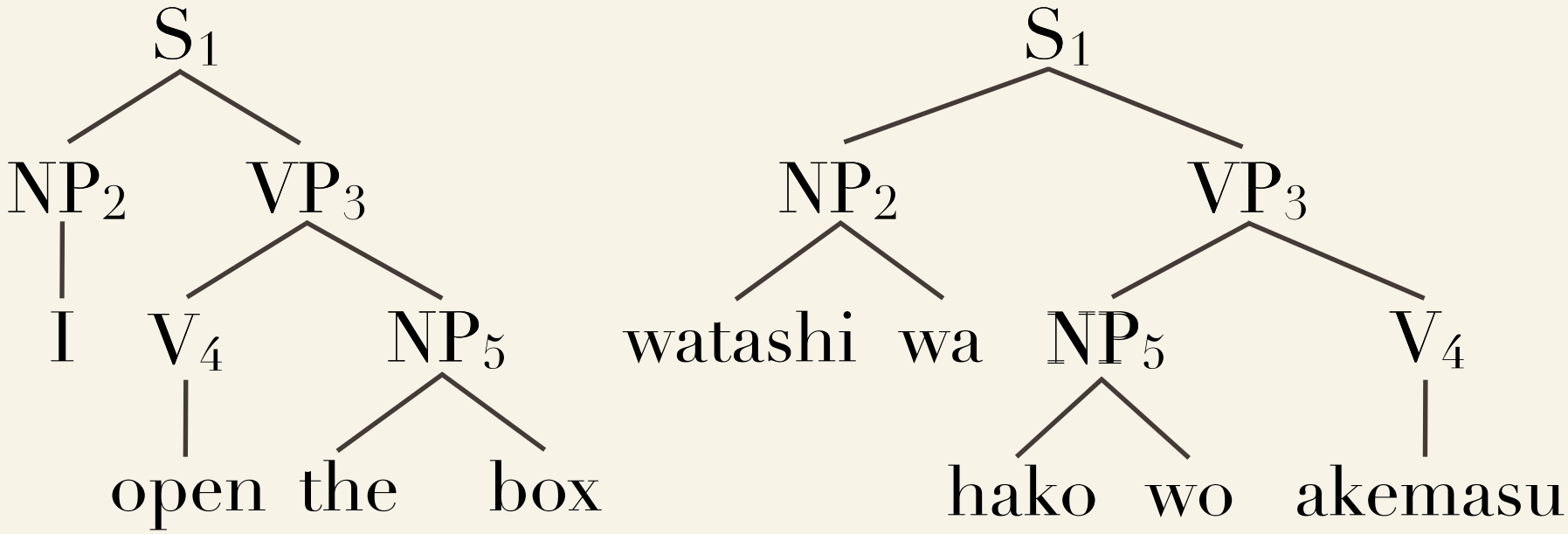
$$NP \rightarrow \text{I, watashi wa}$$

$$NP \rightarrow \text{the box, hako wo}$$

$$VP \rightarrow V_1\ NP_2,\ NP_2\ V_1$$

$$V \rightarrow \text{open, akemasu}$$

Synchronous CFGs



Adding probabilities

$$S \xrightarrow{0.3} NP_1\ VP_2,\ NP_1\ VP_2$$

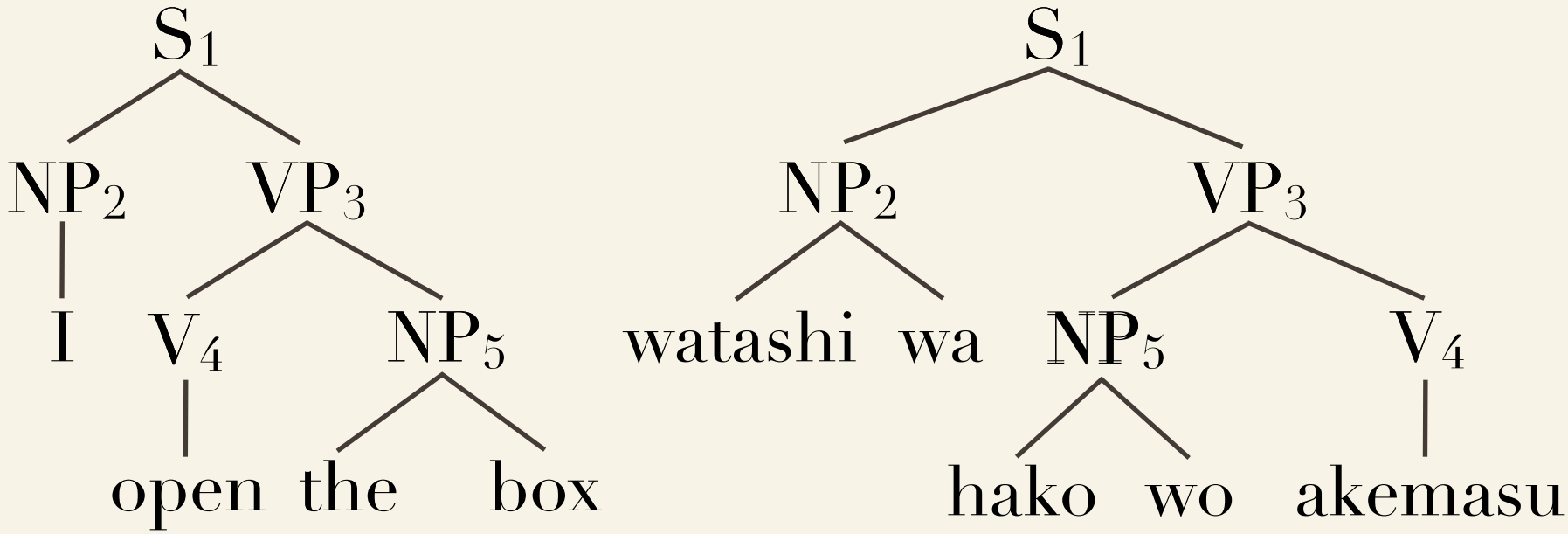
$$NP \xrightarrow{0.1} \text{I, watashi wa}$$

$$NP \xrightarrow{0.6} \text{the box, hako wo}$$

$$VP \xrightarrow{0.5} V_1\ NP_2,\ NP_2\ V_1$$

$$V \xrightarrow{0.2} \text{open, akemasu}$$

Synchronous CFGs



Derivation probability: $0.3 \times 0.1 \times 0.5 \times 0.6 \times 0.2$

Other notations

$$\mathrm{VP} \rightarrow (\mathrm{V}_1\ \mathrm{NP}_2\ ,\ \mathrm{NP}_2\ \mathrm{V}_1)$$

Syntax directed translation schema (Aho and Ullman; Lewis and Stearns)

$$(\mathrm{VP} \rightarrow \mathrm{V}_1\ \mathrm{NP}_2\ ,\ \mathrm{VP} \rightarrow \mathrm{NP}_2\ \mathrm{V}_1)$$

$$\mathrm{VP} \rightarrow \langle\ \mathrm{V}\ \mathrm{NP}\ \rangle$$

Inversion transduction grammar (Wu)

$$\mathrm{VP} \rightarrow \bowtie \begin{matrix}[1,2]\\[2,1]\end{matrix} \begin{pmatrix} \mathrm{V} & \mathrm{NP} \\ \mathrm{V} & \mathrm{NP} \end{pmatrix}$$

Multitext grammar (Melamed)

Properties

Chomsky normal form

$$X \to Y\,Z$$

$$X \to a$$

Chomsky normal form

A → B C D E F *rank* 5

Chomsky normal form

$$A \rightarrow [[B\ \ C]\ D]\ E]\ F$$ rank 5

$$A \rightarrow V1\ F$$

$$V1 \rightarrow V2\ E$$

$$V2 \rightarrow V3\ D$$

$$V3 \rightarrow B\ C$$

rank 2

A hierarchy of synchronous CFGs

$$\text{1-CFG} \subsetneq \text{2-CFG} = \text{3-CFG} = \text{4-CFG} = \ldots$$

$$\text{1-SCFG} \subsetneq \text{2-SCFG} = \text{3-SCFG} \subsetneq \text{4-SCFG} \subsetneq \ldots$$

$$\diagdown \quad \diagup$$

ITG

(Wu, 1997)

Synchronous CNF?

$$A \rightarrow (B_1\ C_2\ D_3\ ,\ C_2\ D_3\ B_1) \qquad \text{rank } 3$$

Synchronous CNF?

$$A \rightarrow (B_1\ [C_2\ \ D_3], [C_2\ \ D_3]\ B_1)$$ rank 3

$$A \rightarrow (B_1\ V1_2\ , V1_2\ B_1)$$
$$V1 \rightarrow (C_1\ D_2\ \ , C_1\ D_2)$$ rank 2

Synchronous CNF?

$A \rightarrow (B_1\ C_2\ D_3\ E_4,\ C_2\ E_4\ B_1\ D_3)$ rank 4

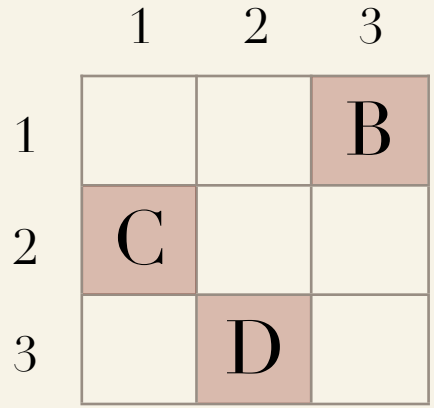
$A \rightarrow ([B_1\ C_2]\ D_3\ E_4,\ [C_2\ E_4\ B_1]\ D_3)$

$A \rightarrow (B_1\ [C_2\ D_3]\ E_4,\ [C_2\ E_4\ B_1\ D_3])$

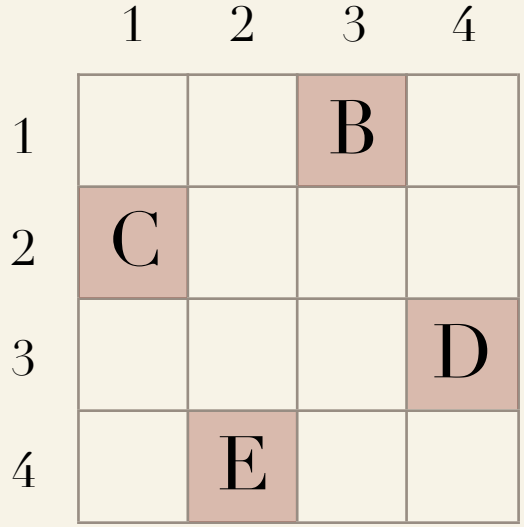
$A \rightarrow (B_1\ C_2\ [D_3\ E_4],\ C_2\ [E_4\ B_1\ D_3])$

Synchronous CNF?

$A \rightarrow (B_1\ C_2\ \ D_3, C_2\ D_3\ B_1)$



$A \rightarrow (B_1\ C_2\ D_3\ E_4, C_2\ E_4\ B_1\ D_3)$



A hierarchy of synchronous CFGs

$$\text{1-CFG} \subsetneq \text{2-CFG} = \text{3-CFG} = \text{4-CFG} = \ldots$$

$$\text{1-SCFG} \subsetneq \text{2-SCFG} = \text{3-SCFG} \subsetneq \text{4-SCFG} \subsetneq \ldots$$

$$\diagdown \quad \diagup$$

ITG

(Wu, 1997)

Algorithms

Overview

- Translation
- Bitext parsing

Review: CKY

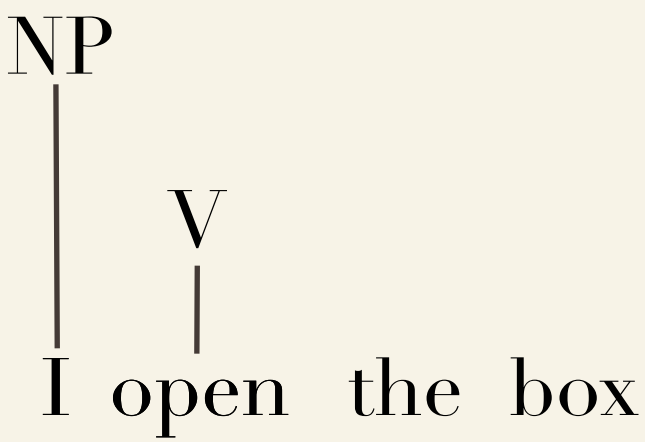
S → NP VP

NP → I

NP → the box

VP → V NP

V → open



Review: CKY

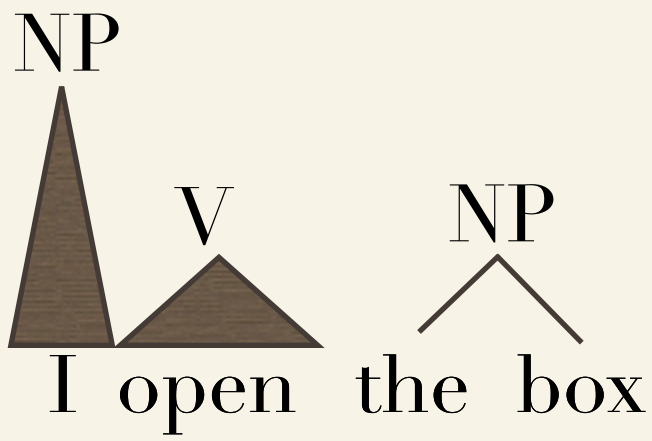
S → NP VP

NP → I

NP → the box

VP → V NP

V → open



Review: CKY

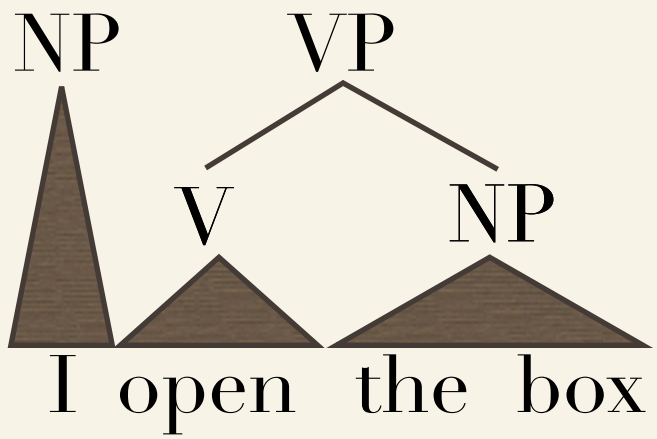
S → NP VP

NP → I

NP → the box

VP → V NP

V → open



Review: CKY

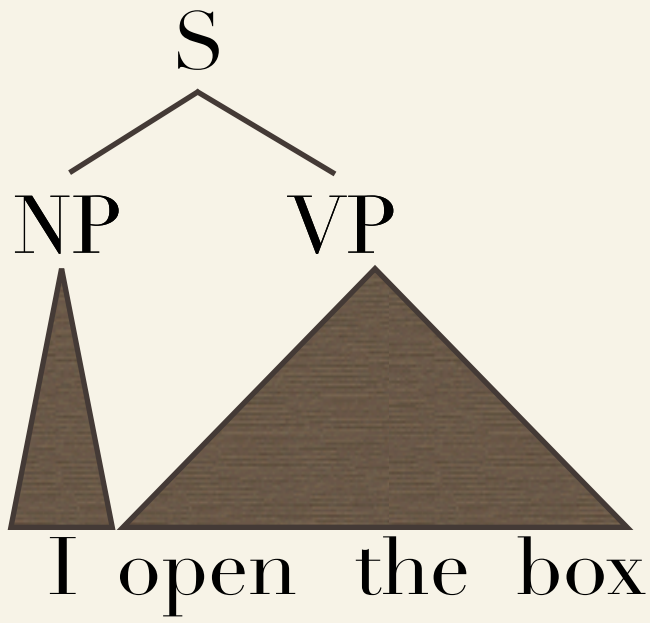
S → NP VP

NP → I

NP → the box

VP → V NP

V → open



Review: CKY

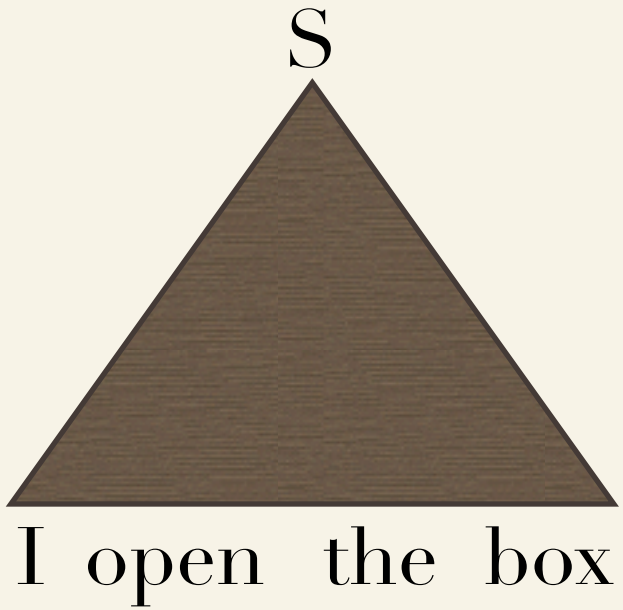
S → NP VP

NP → I

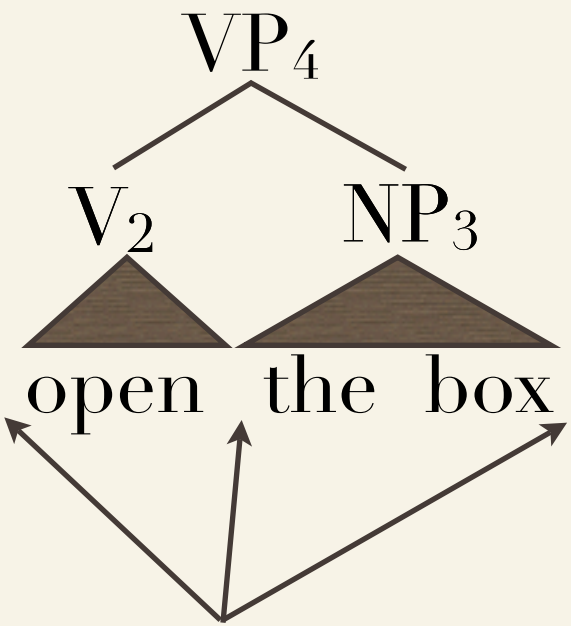
NP → the box

VP → V NP

V → open

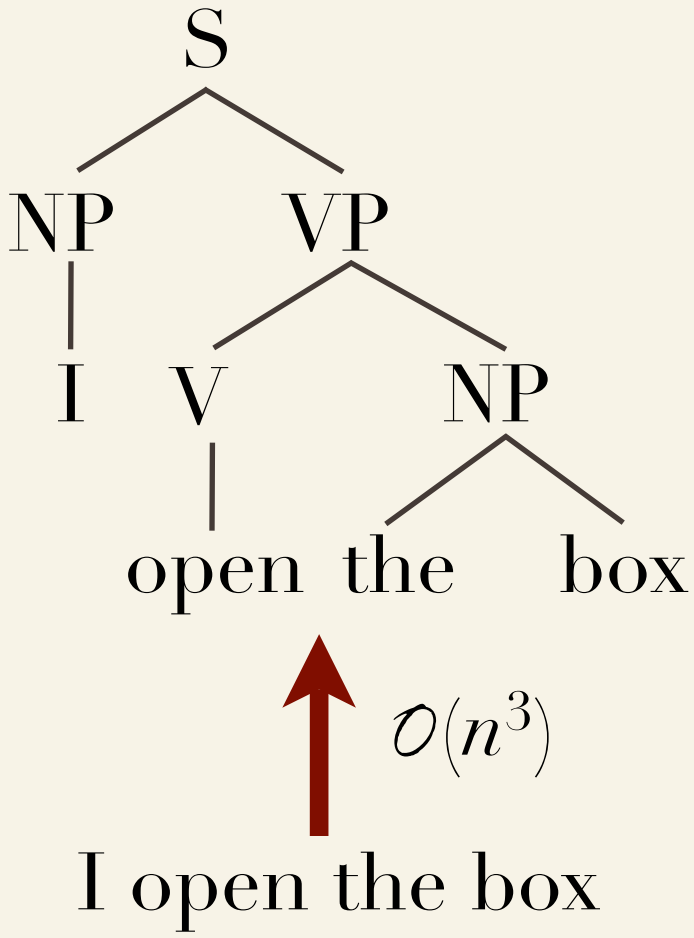


Review: CKY

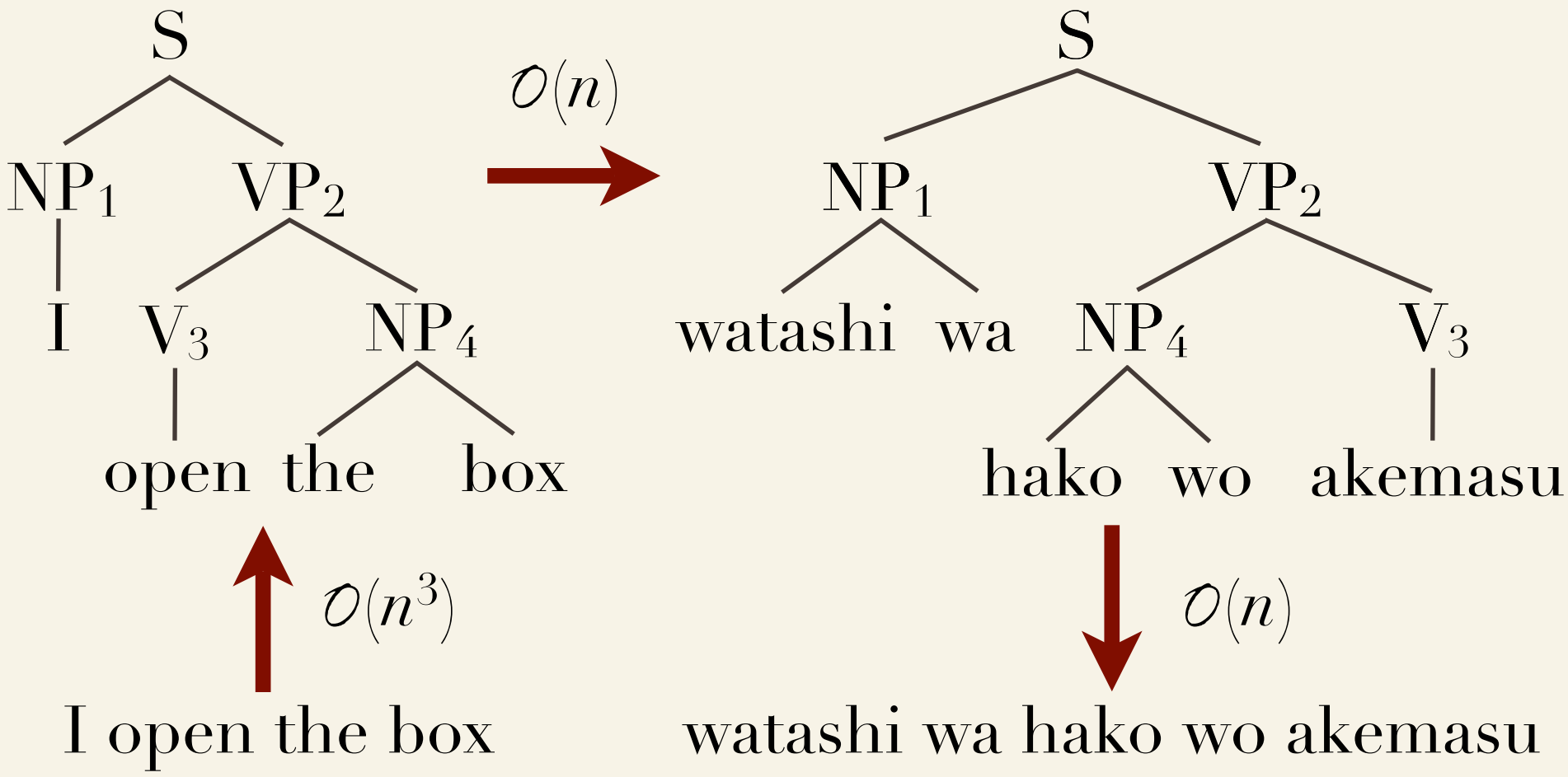


$\mathcal{O}(n^3)$ ways of matching

Translation

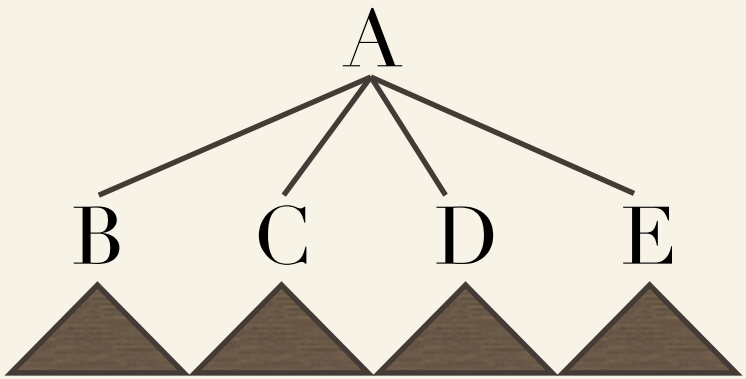


Translation



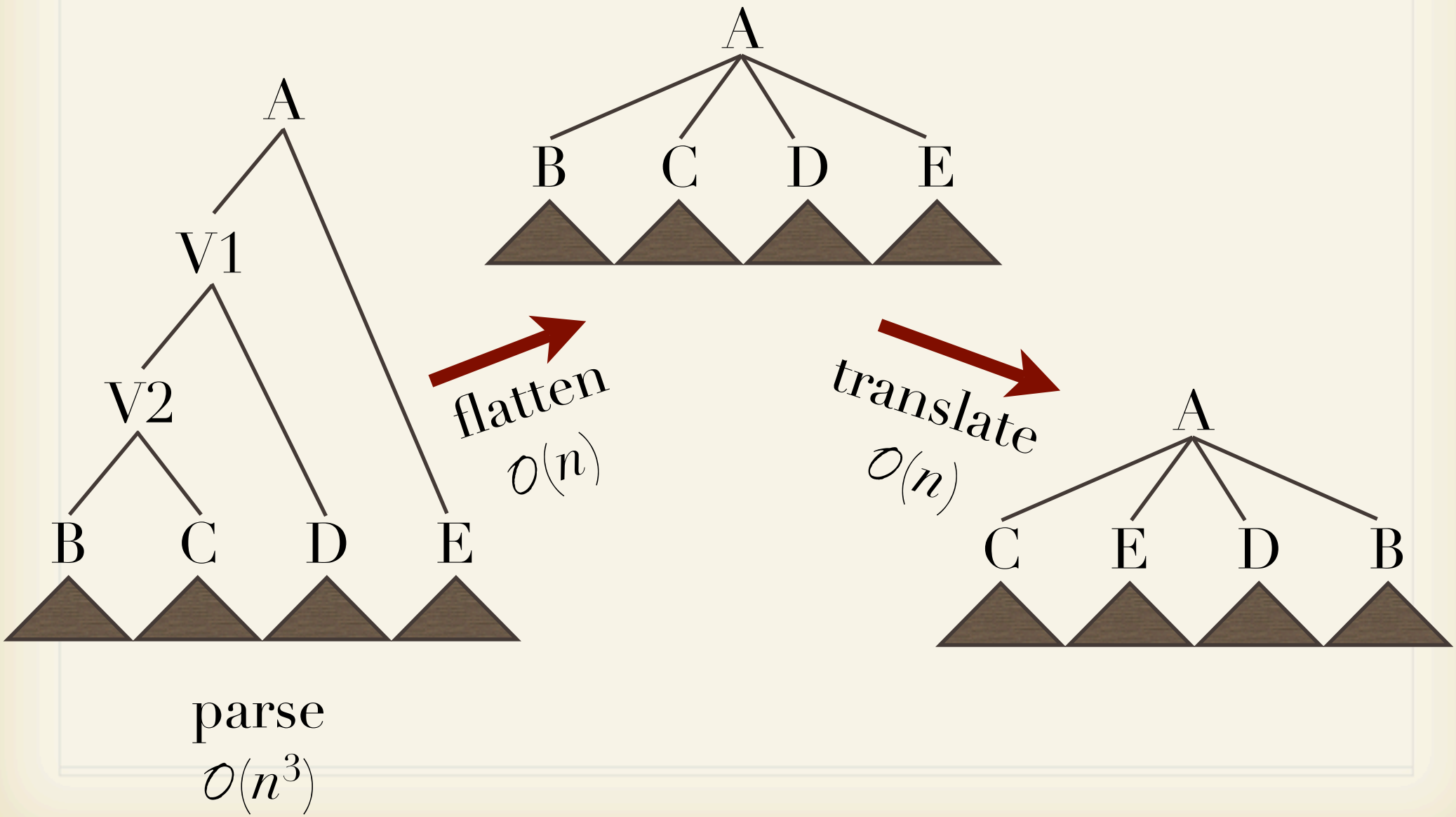
Translation

What about…

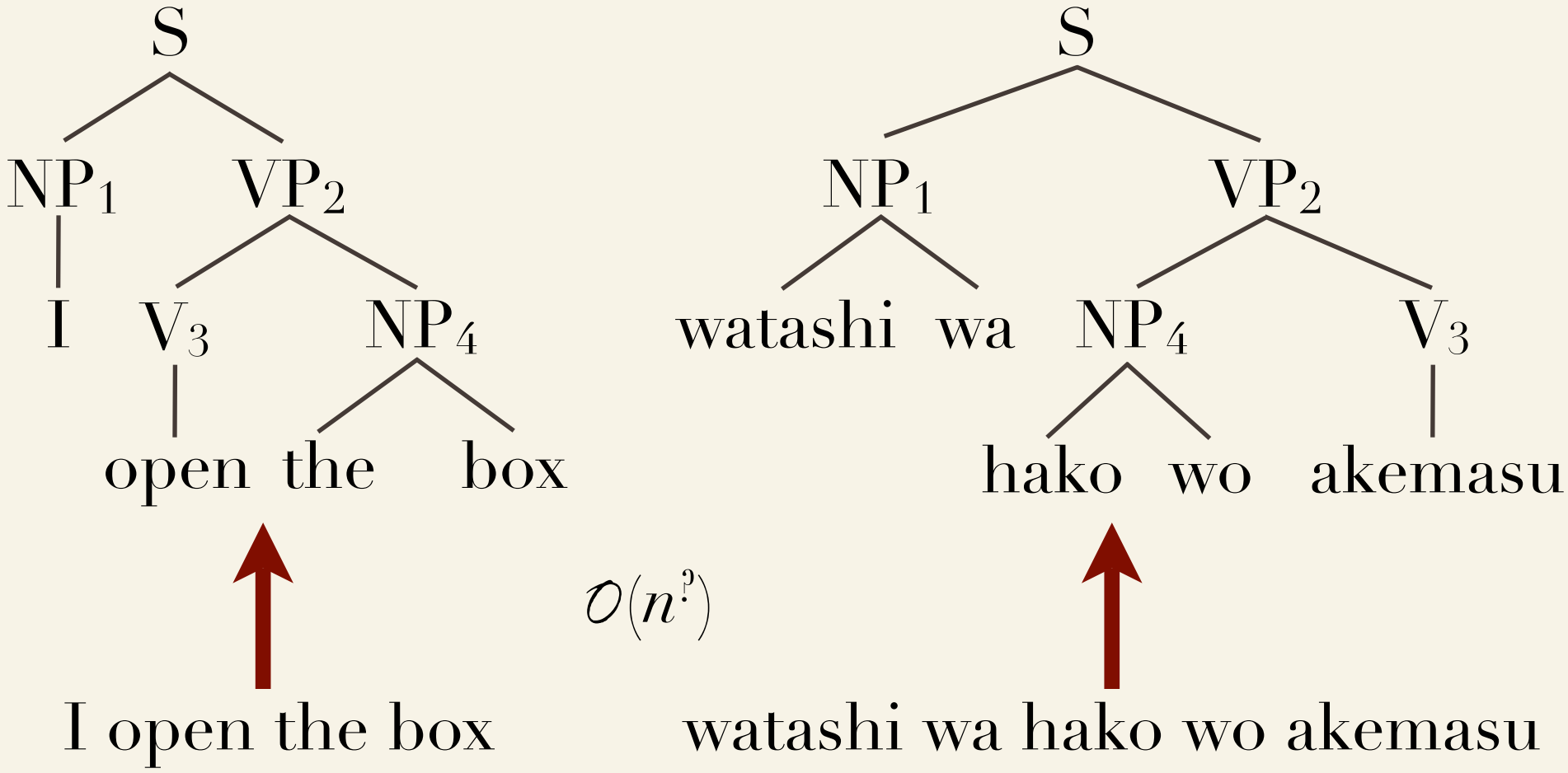


$\mathcal{O}(n^5)$ ways of combining?

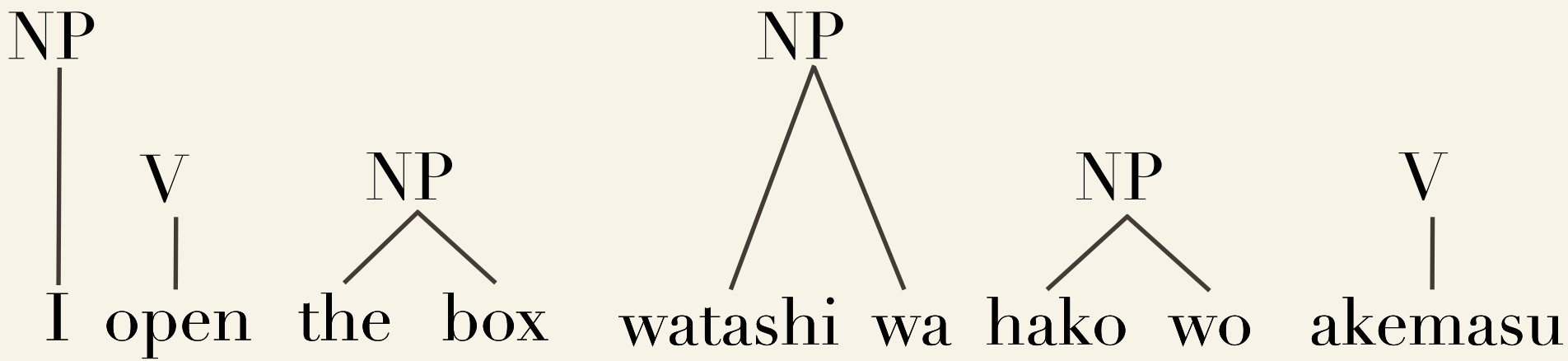
Translation



Bitext parsing

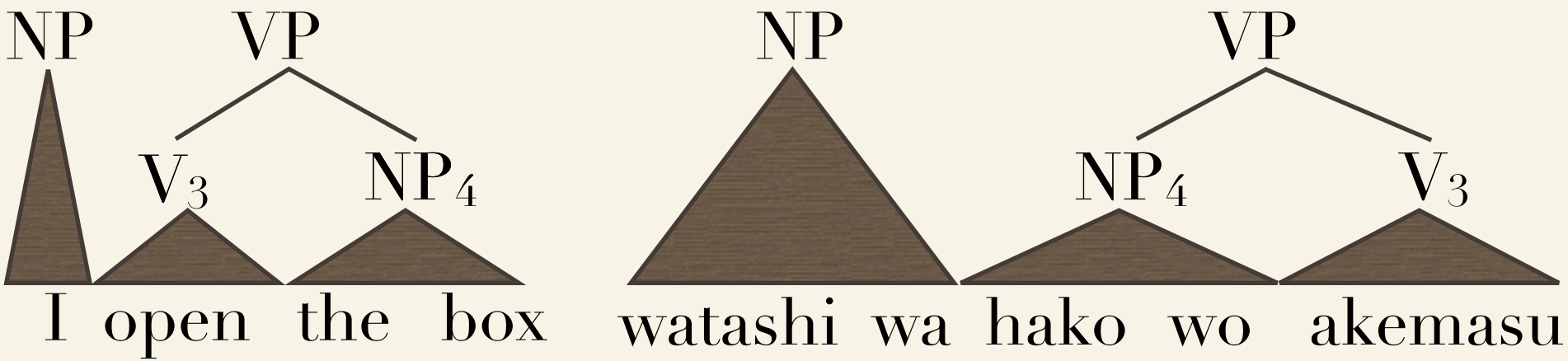


Bitext parsing

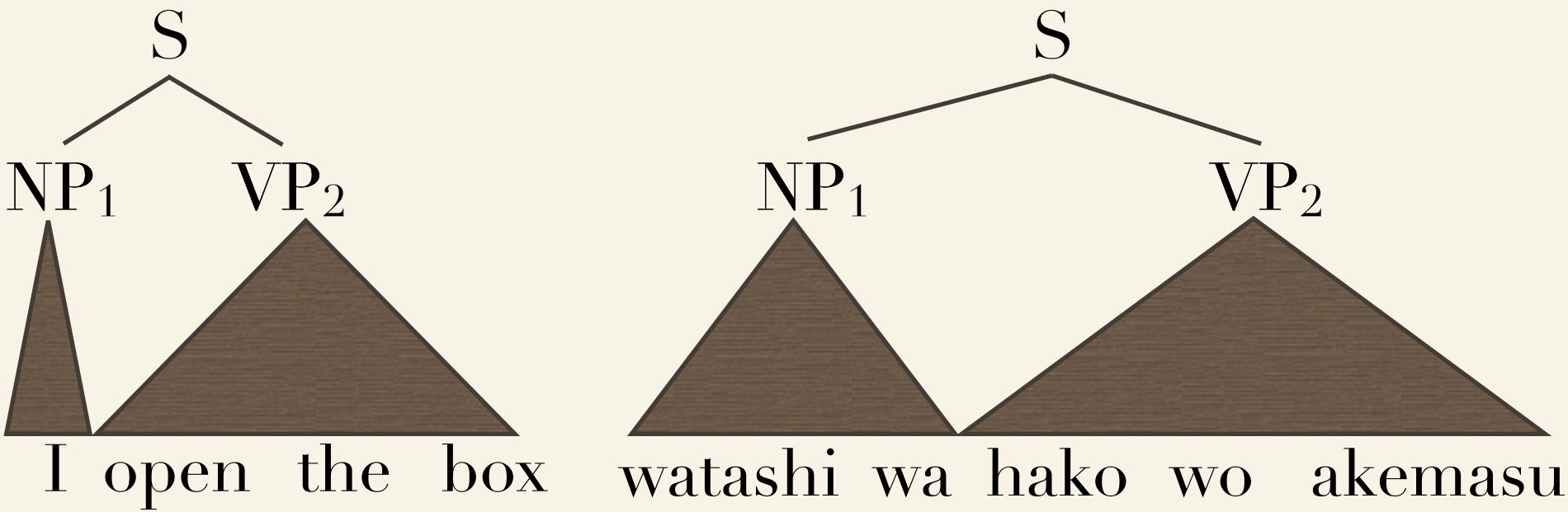


Consider rank-2 synchronous CFGs for now

Bitext parsing

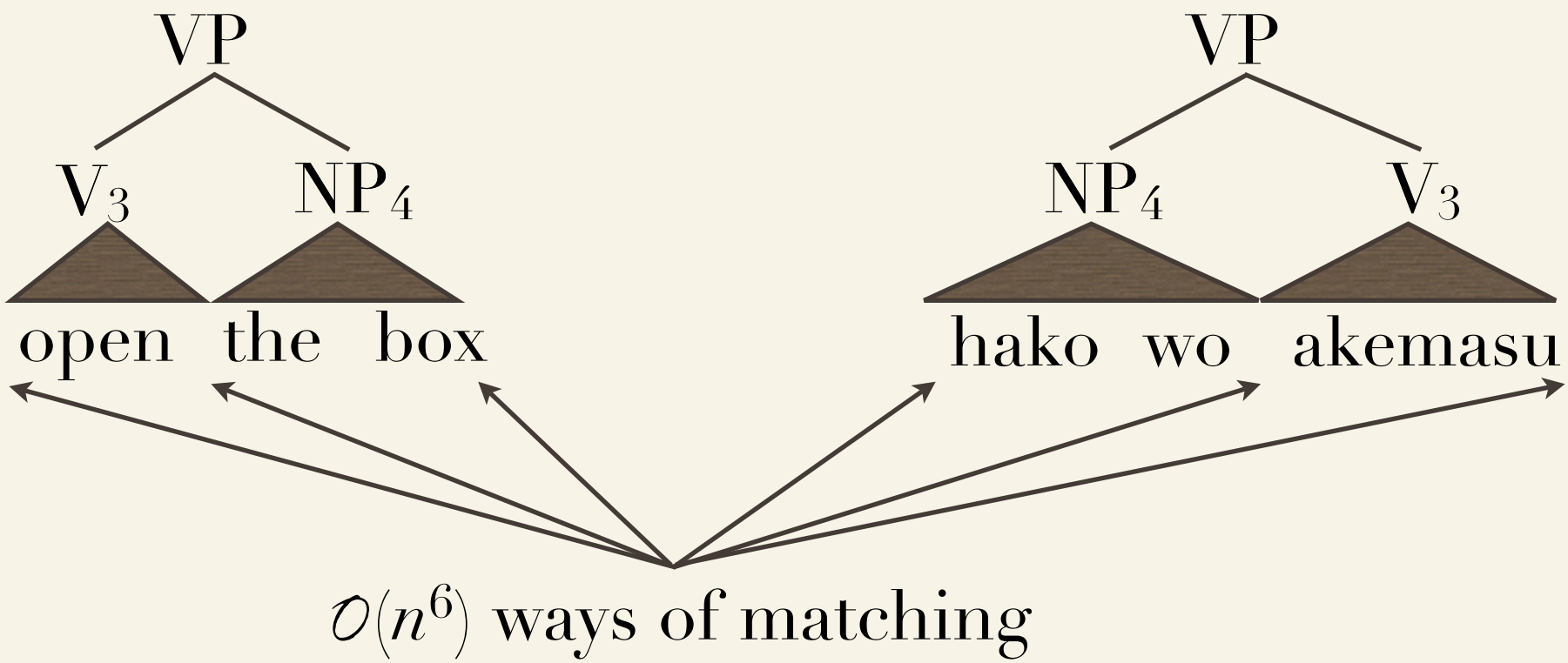


Bitext parsing

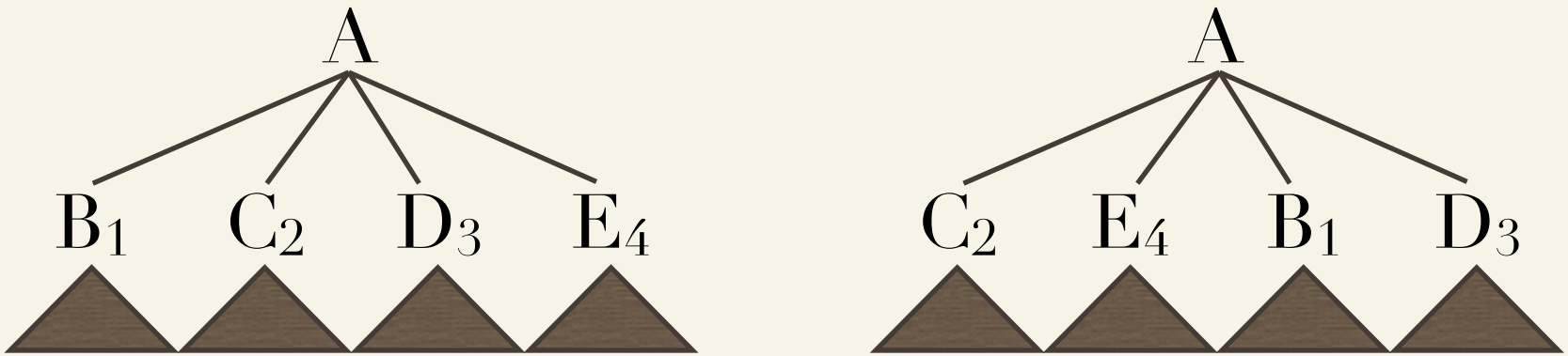


Bitext parsing

Bitext parsing



Bitext parsing



$\mathcal{O}(n^{10})$ ways of combining!

Algorithms

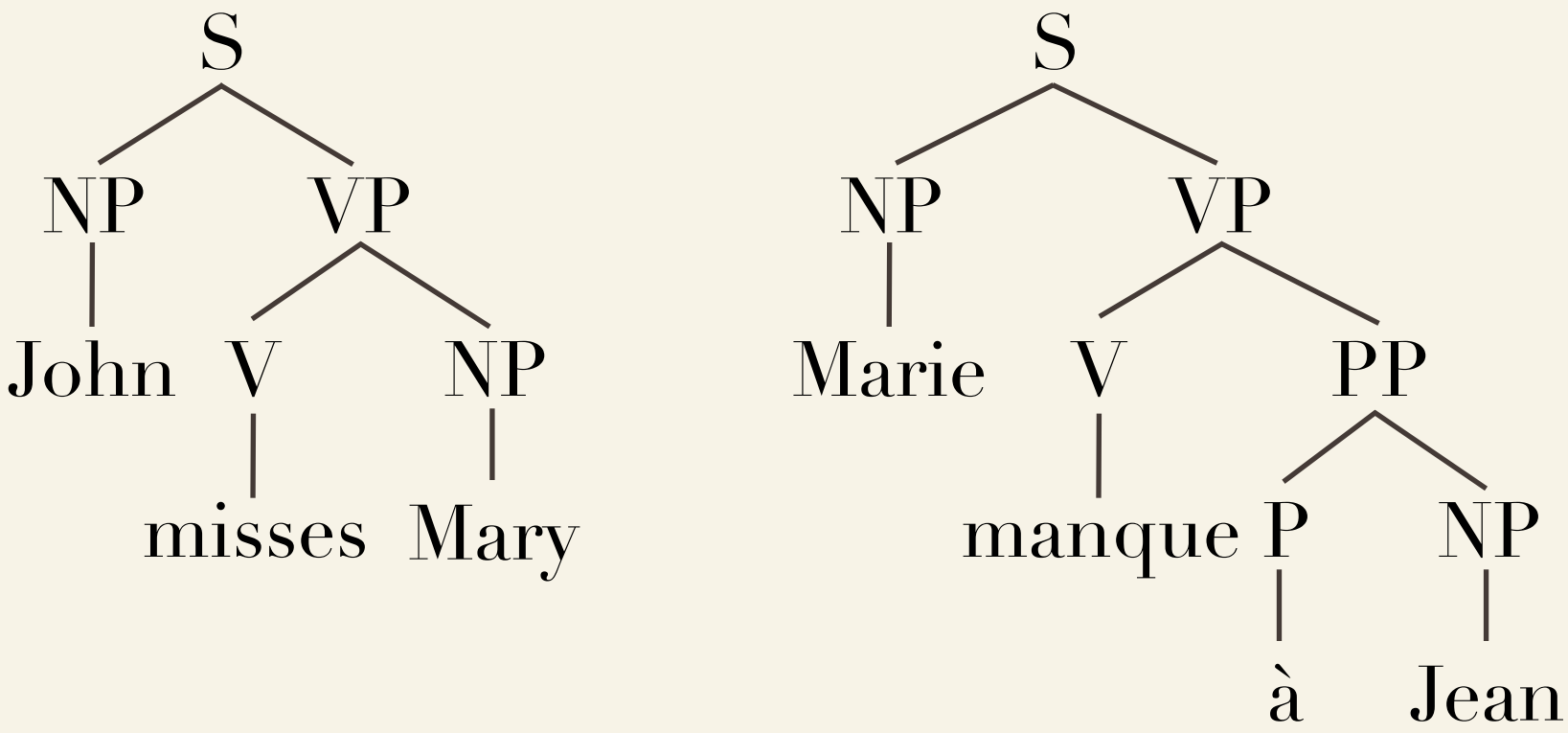
- Translation: easy
- Bitext parsing: polynomial in n but worst-case exponential in rank

Algorithms

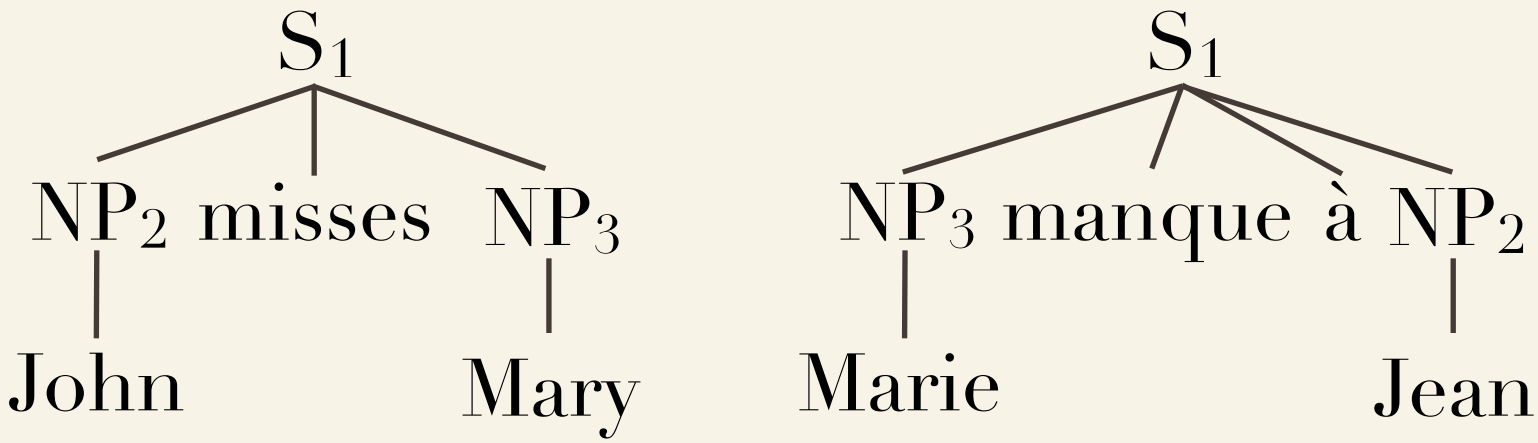
- Translation with an n-gram language model
  - Offline rescoring
  - Intersect grammar and LM (Wu 1996; Huang et al. 2005): slower
  - Hybrid approaches (Chiang 2005; Zollman and Venugopal 2006)

Extensions

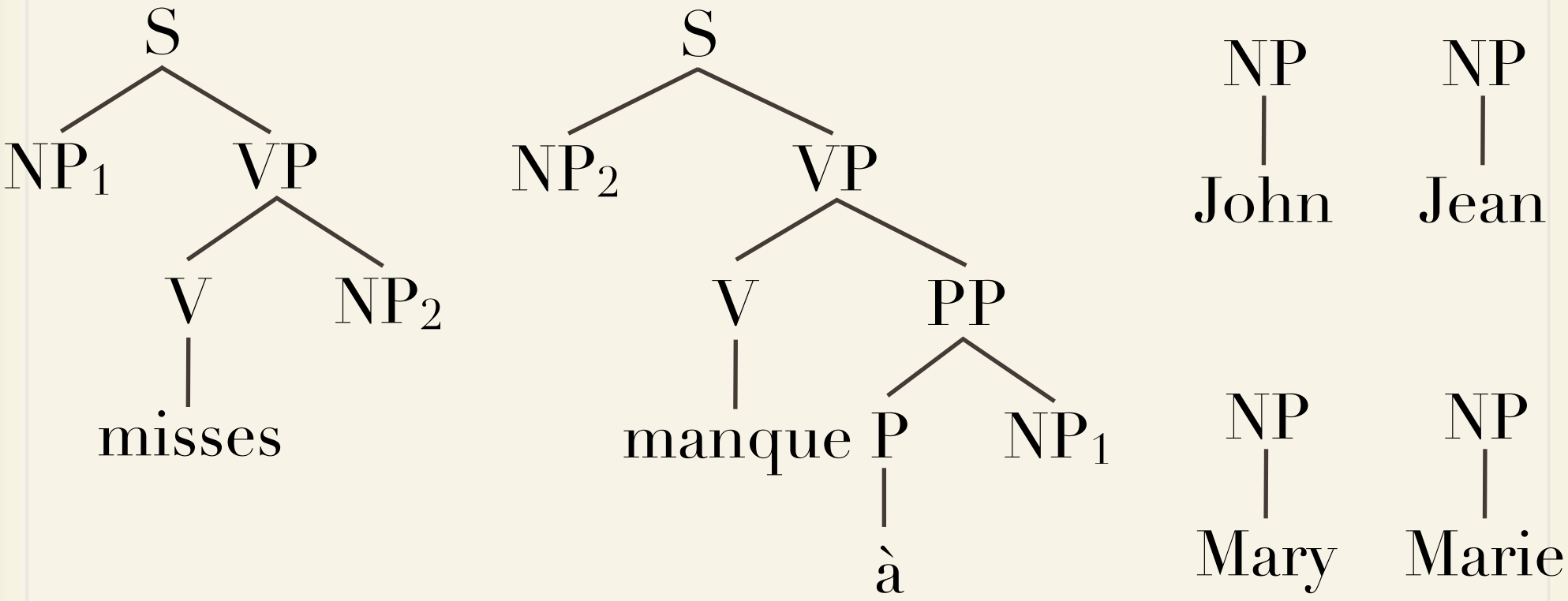
Limitations of synchronous CFGs



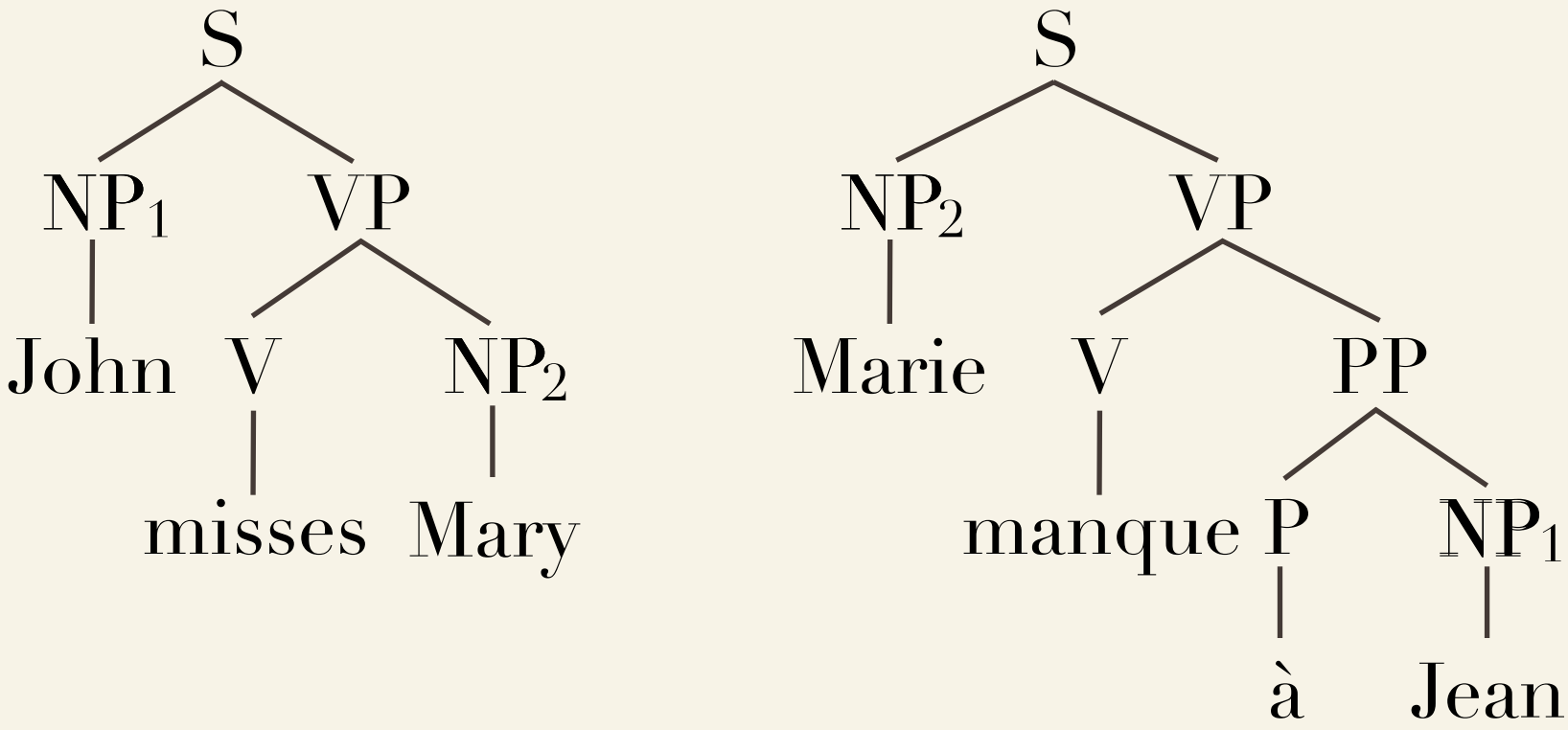
One solution



Synchronous tree substitution grammars

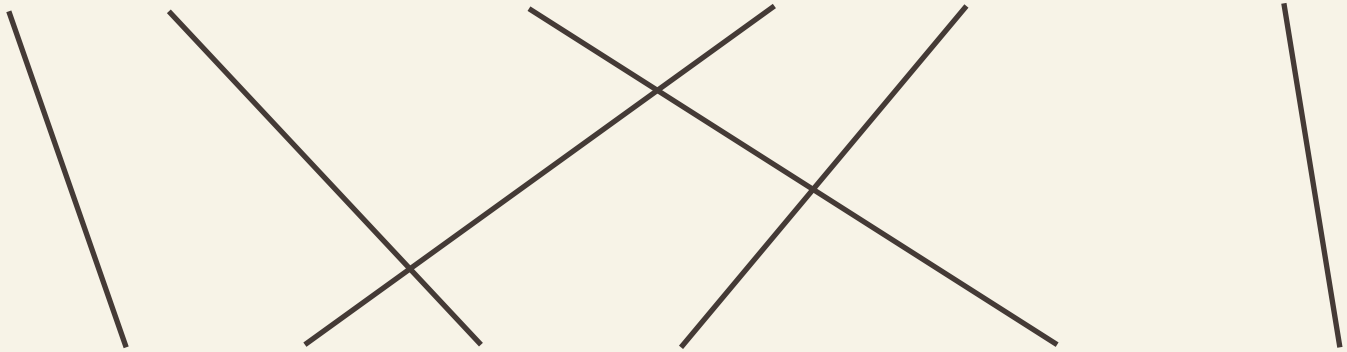


Synchronous tree substitution grammars



Limitations of synchronous TSGs

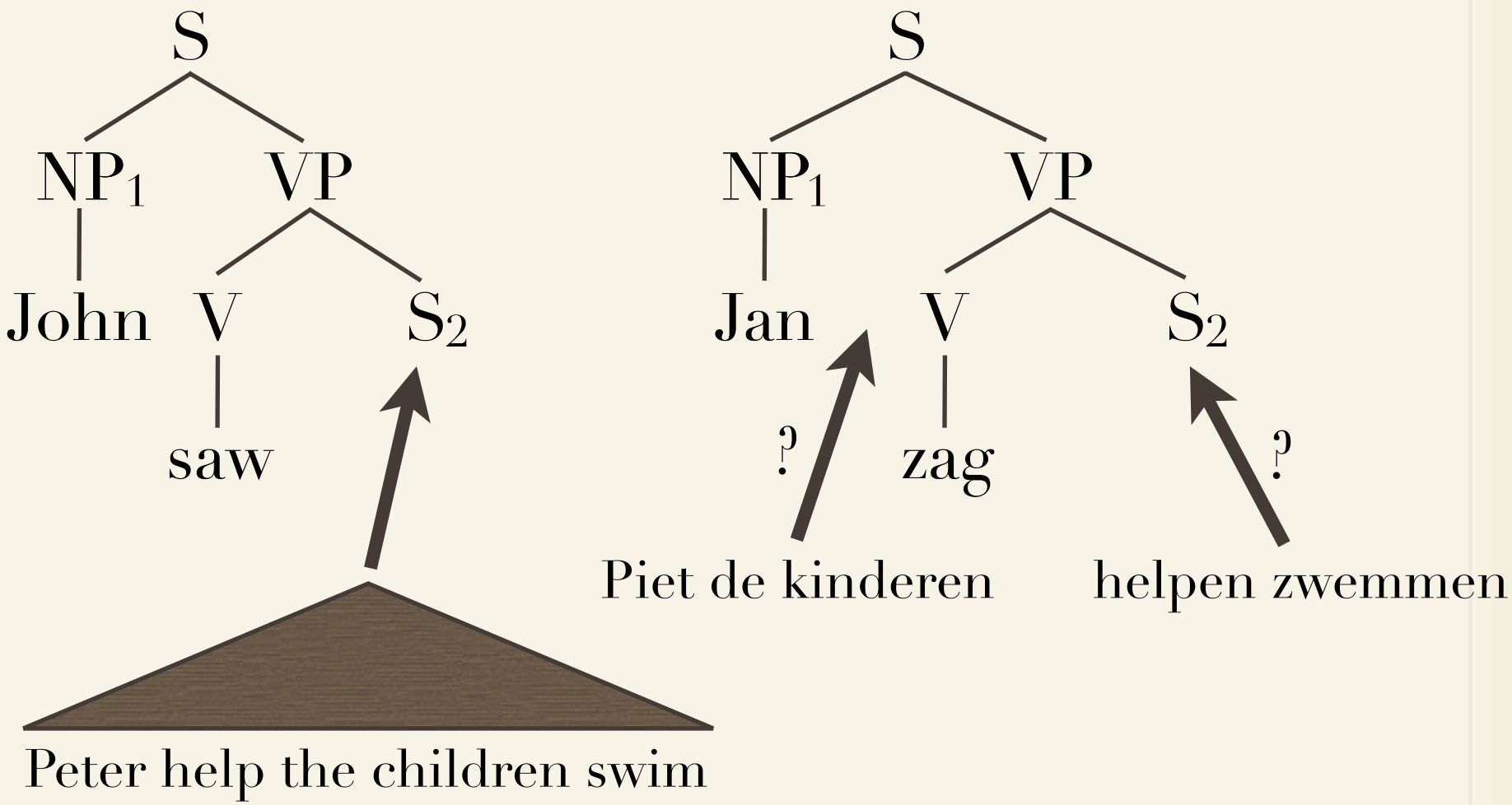
…dat Jan Piet de kinderen zag helpen zwemmen



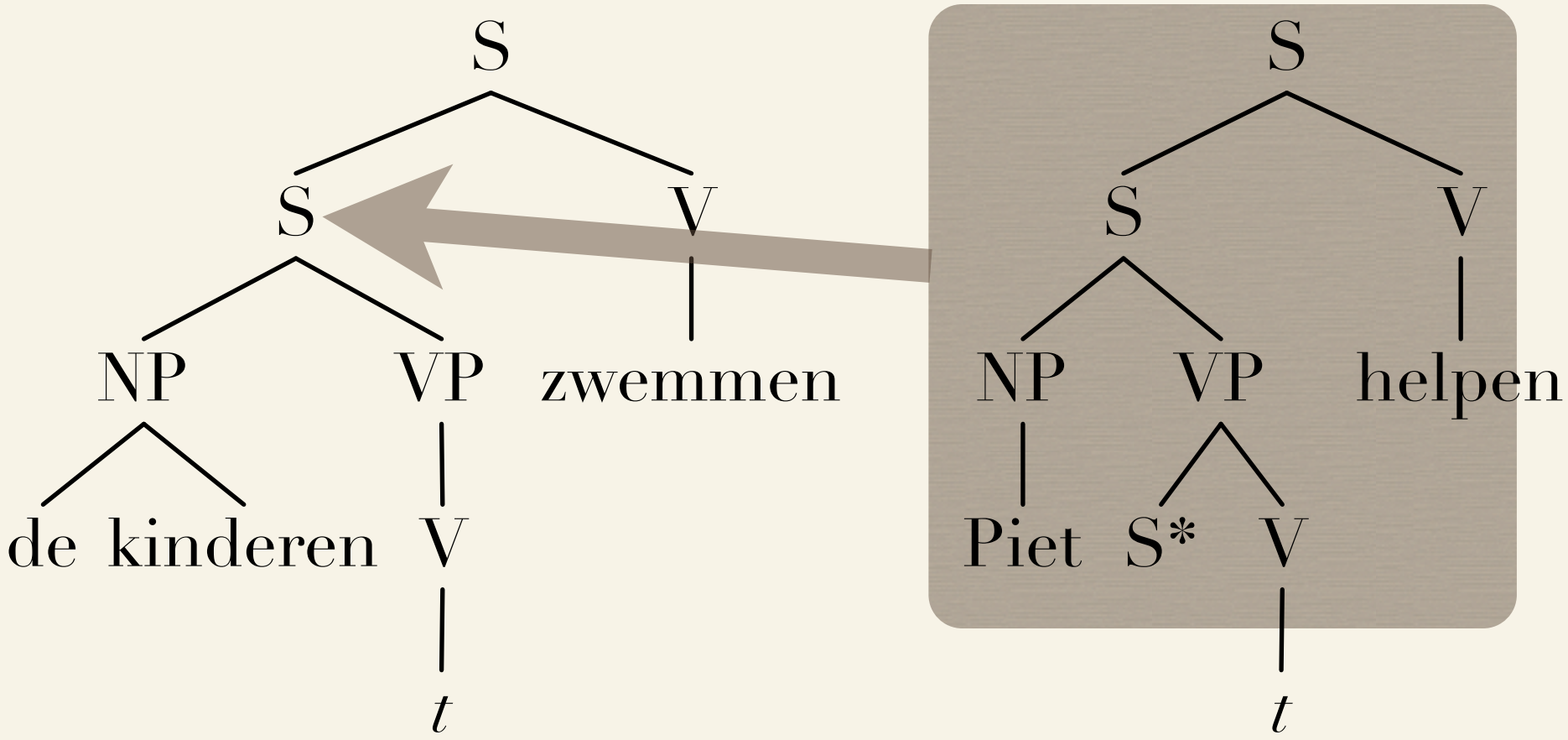
…that John saw Peter help the children swim

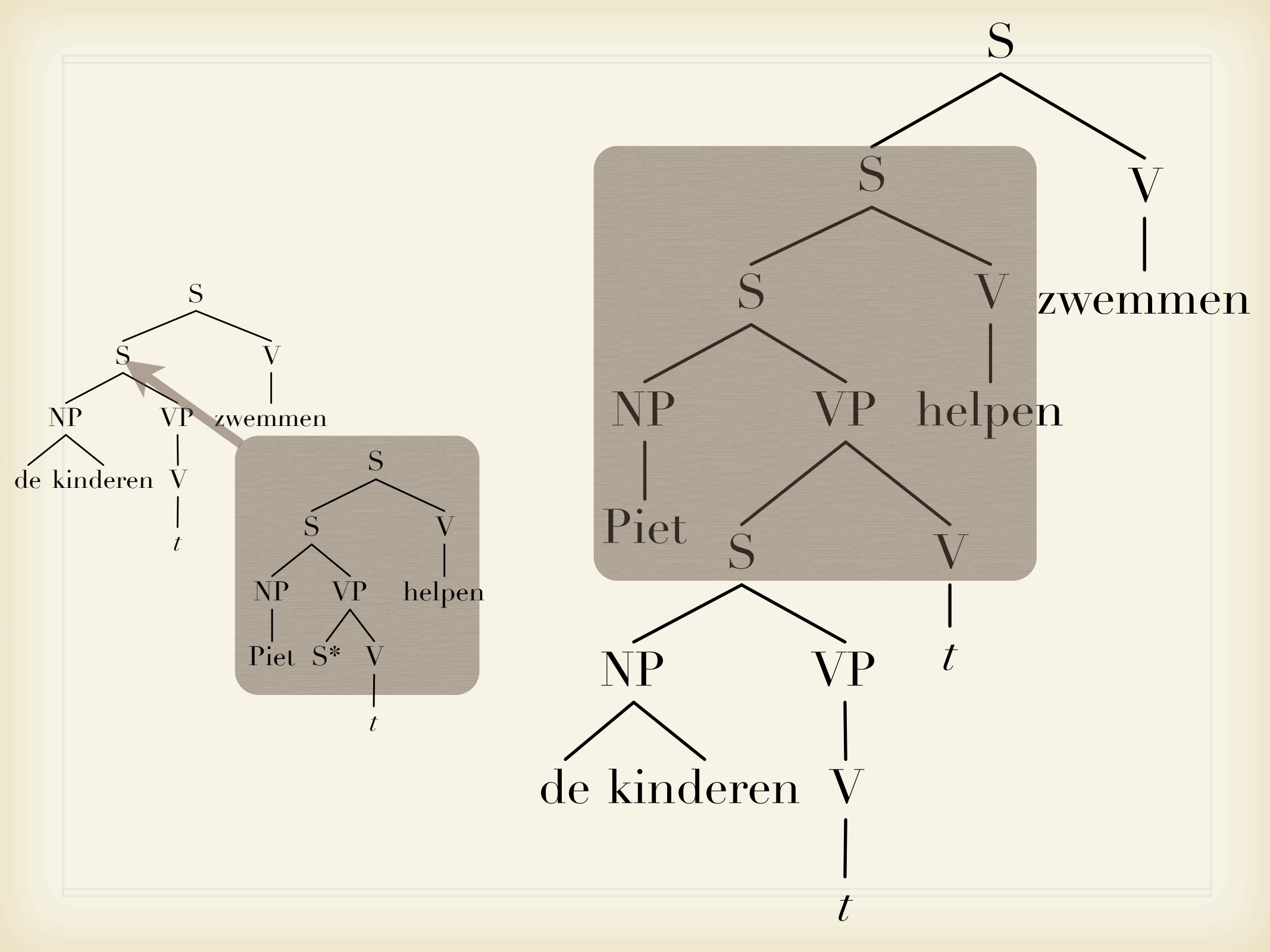
This pattern extends to n nouns and n verbs

Limitations of synchronous TSGs

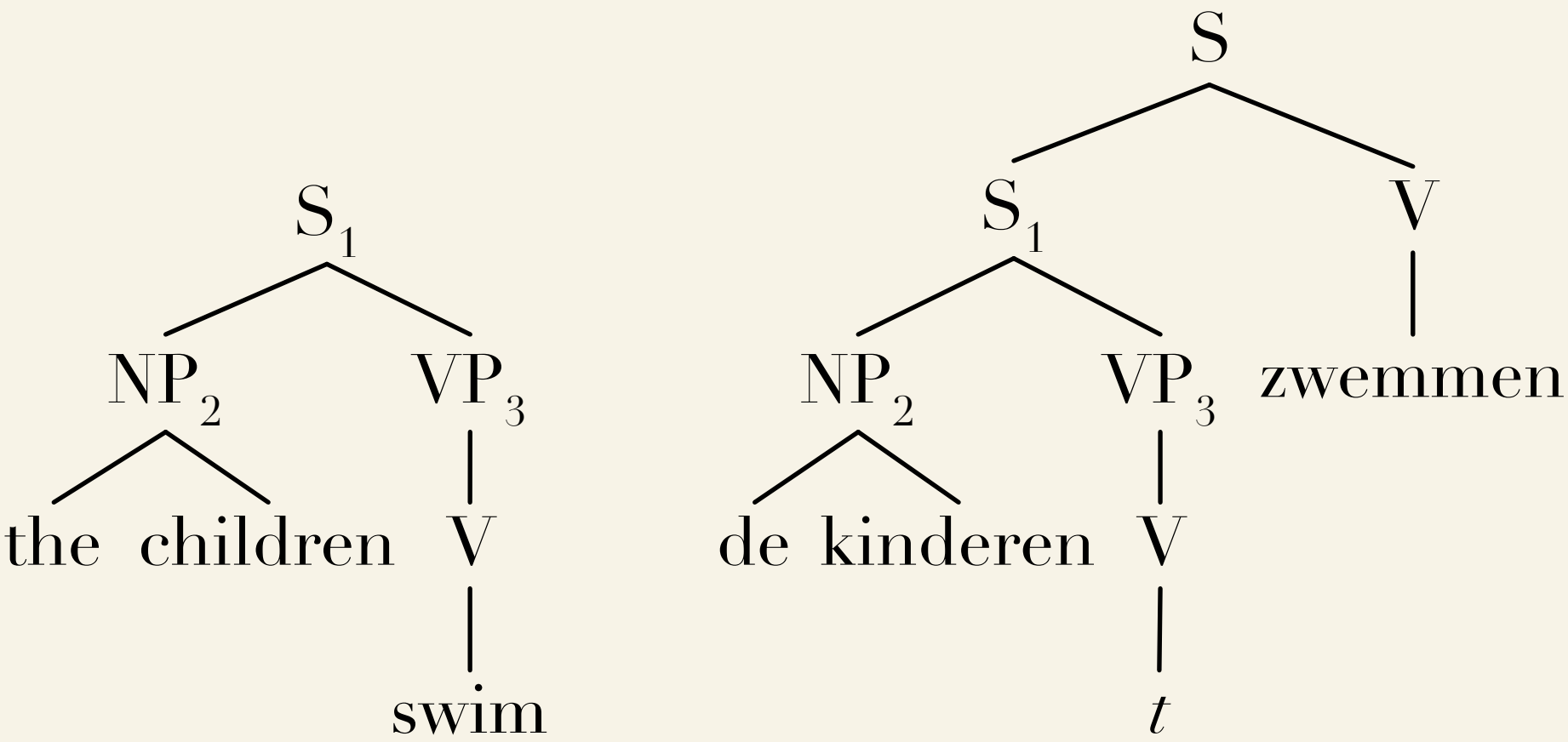


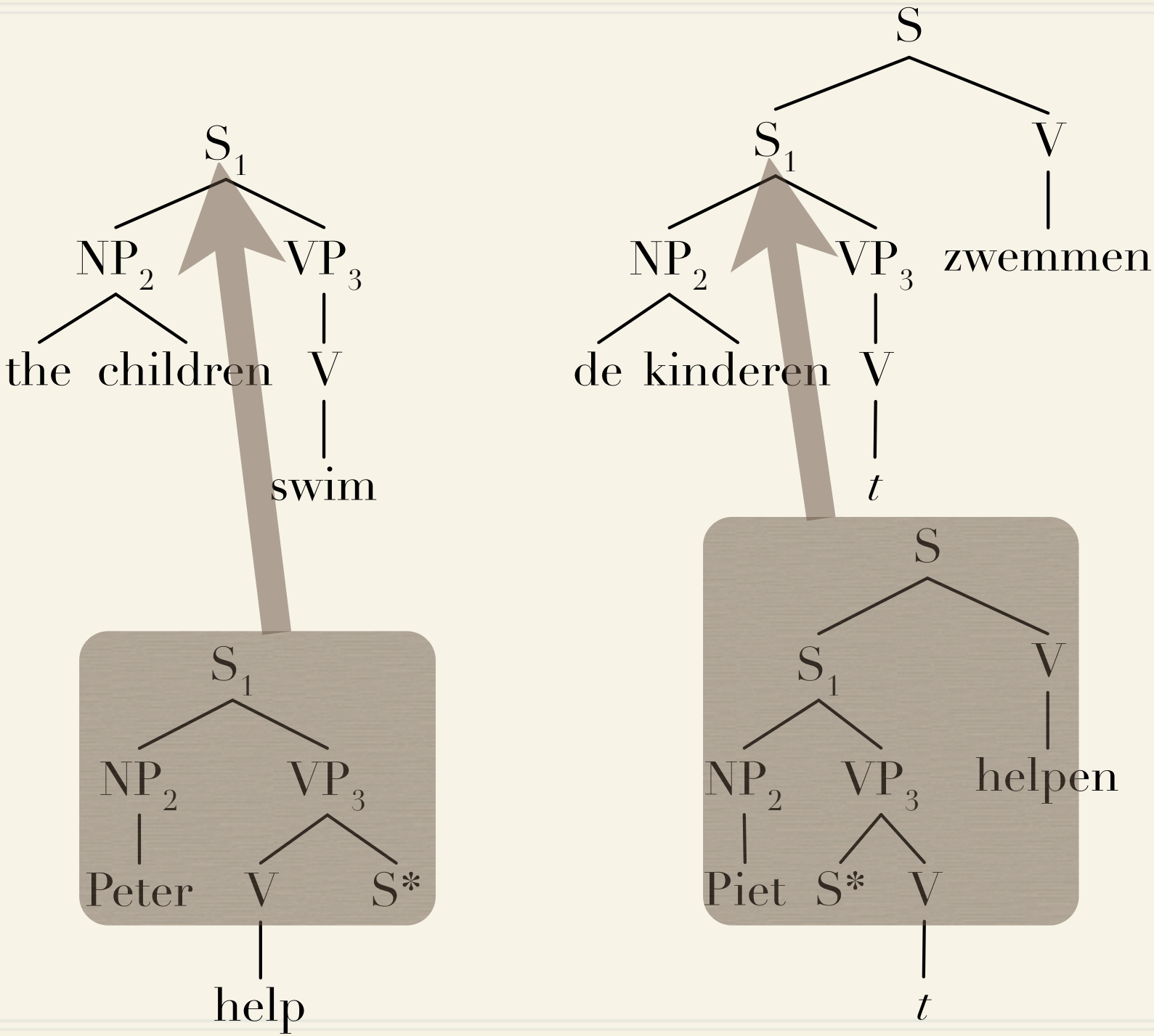
Tree-adjoining grammar

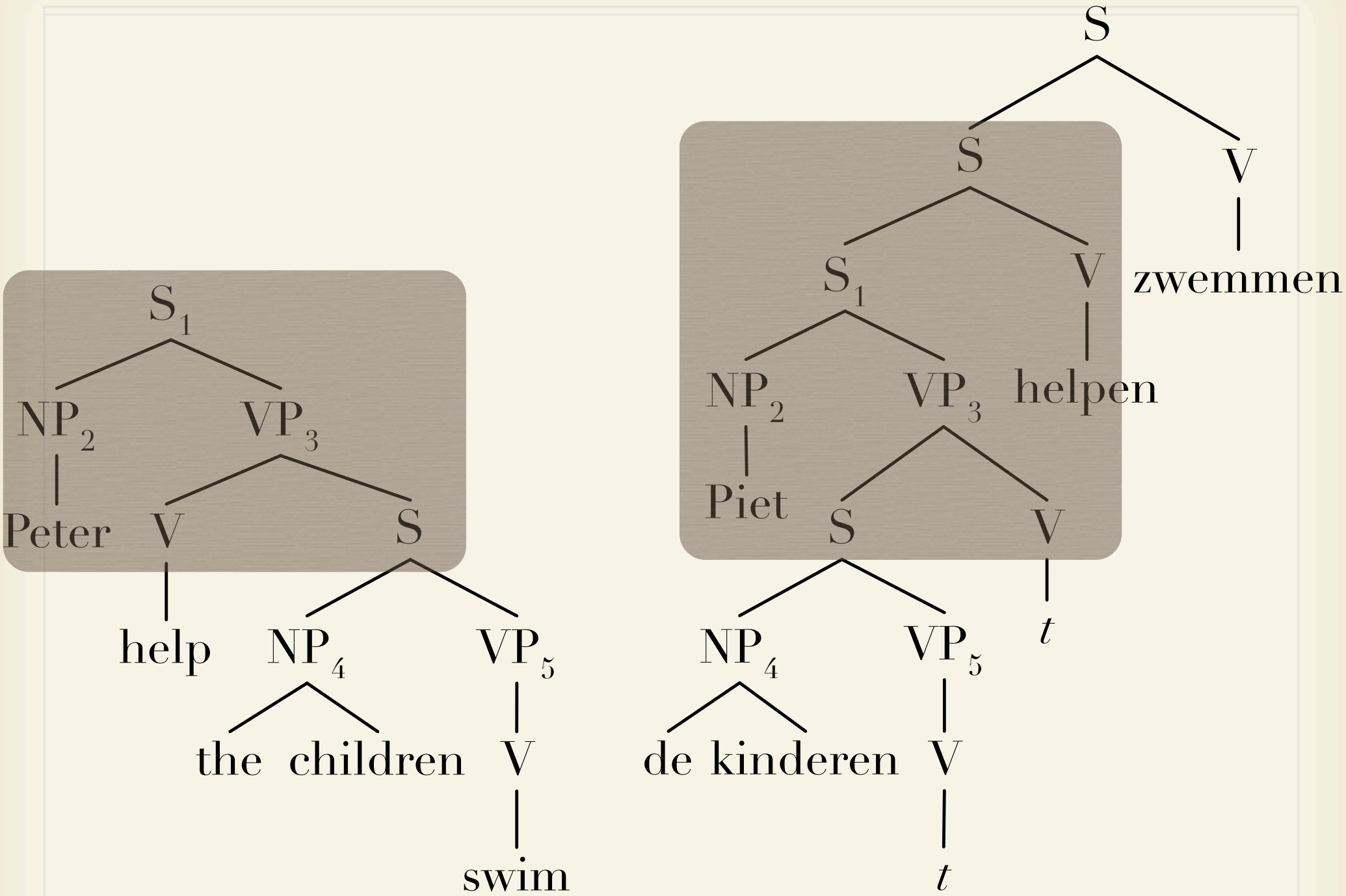


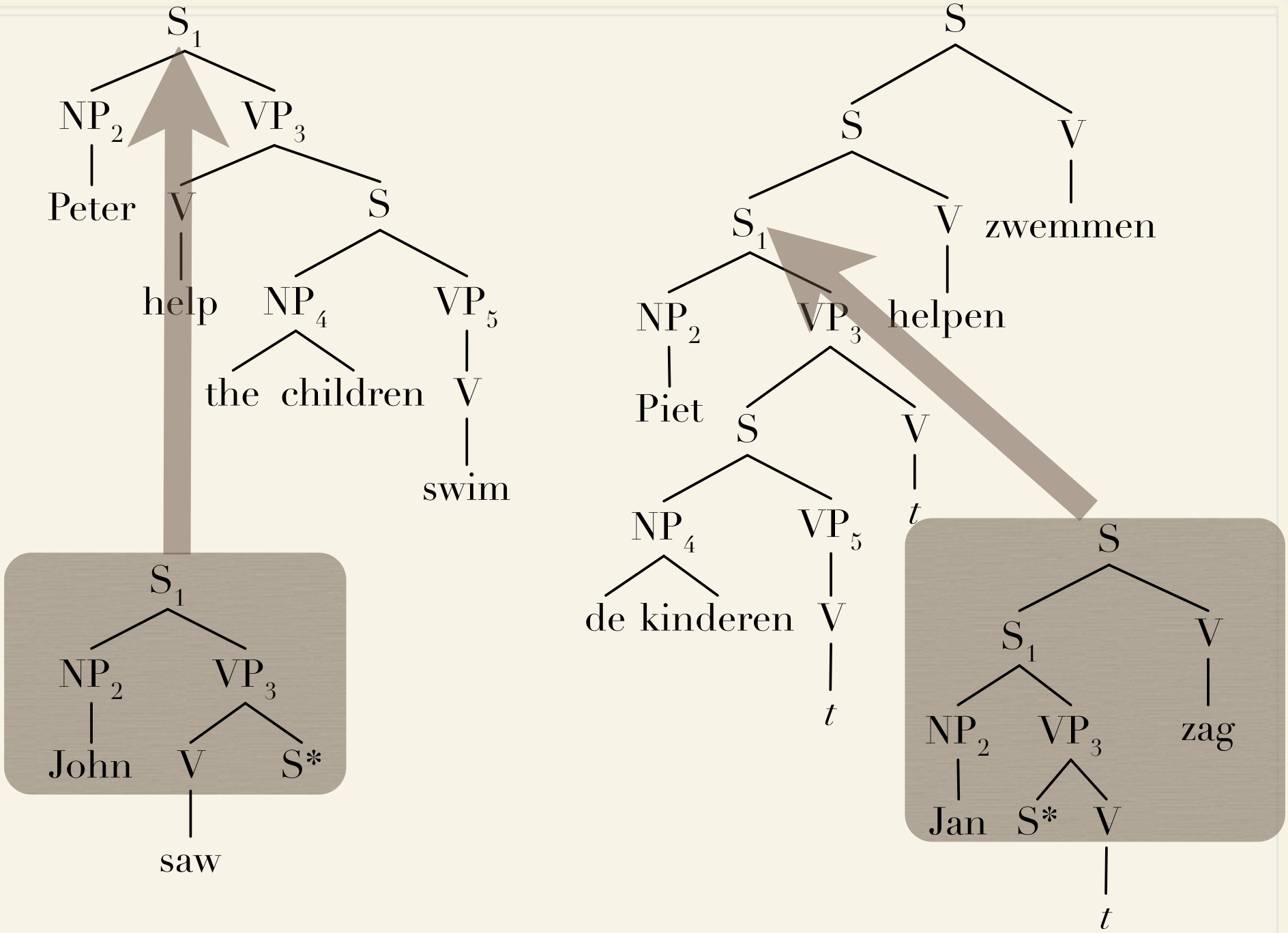

S
S
V
NP
VP
zwemmen
de kinderen
V
t
S
V
NP
VP
helpen
Piet
S*
V
t
S
S
V
zwemmen
NP
VP
helpen
Piet
S
V
t
NP
VP
de kinderen
V
t

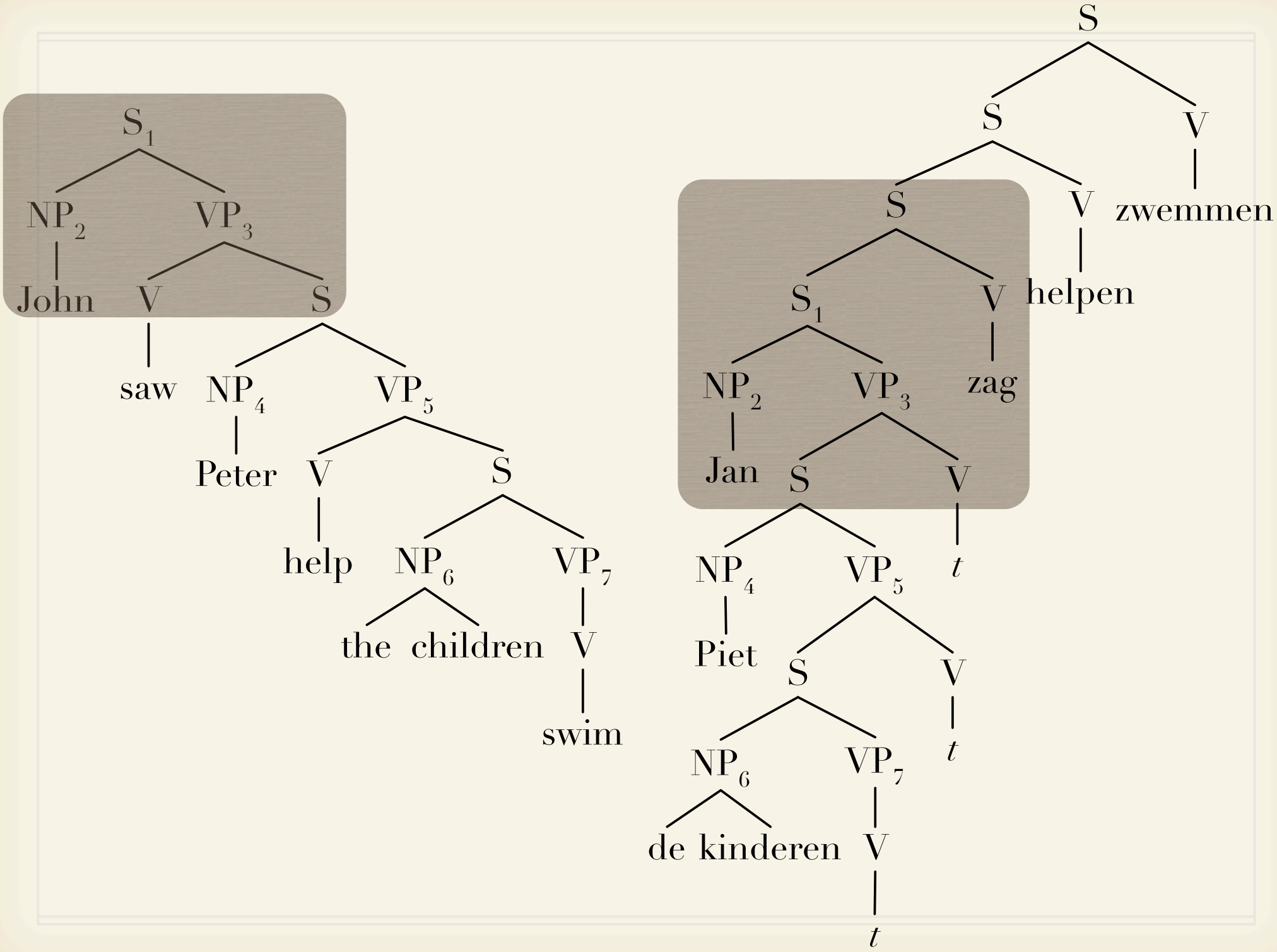
Synchronous TAG




S1
NP2
VP3
the children
V
swim
S1
NP2
VP3
Peter
V
S*
help
S
S1
V
NP2
VP3
zwemmen
de kinderen
V
t
S
S1
V
NP2
VP3
helpen
Piet
S*
V
t


S
S
V
zwemmen
S1
V
helpen
NP2
VP3
Piet
S
V
t
NP4
VP5
de kinderen
V
t
S1
NP2
VP3
Peter
V
S
help
NP4
VP5
the children
V
swim


S1
NP2
VP3
Peter
V
S
help
NP4
VP5
the children
V
swim
S1
NP2
VP3
John
V
S*
saw
S
S
V
S1
V
zwemmen
NP2
VP3
helpen
Piet
S
V
t
NP4
VP5
de kinderen
V
t
S
S1
V
NP2
VP3
zag
Jan
S*
V
t


S
S
V
S
V
zwemmen
S1
V
helpen
NP2
VP3
zag
Jan
S
V
NP4
VP5
t
Piet
S
V
t
NP6
VP7
de kinderen
V
t
S1
NP2
VP3
John
V
S
saw
NP4
VP5
Peter
V
S
help
NP6
VP7
the children
V
swim

Summary

- Synchronous grammars are useful for various tasks including translation
- Some rules "in the wild" (Chiang, 2005):

$$X \rightarrow (\text{de}, \text{'s})$$

$$X \rightarrow (X_1 \text{ de } X_2, \text{the } X_2 \text{ of } X_1)$$

$$X \rightarrow (X_1 \text{ de } X_2, \text{the } X_2 \text{ that})$$

$$X \rightarrow (\text{zai } X_1 \text{ xia}, \text{under } X_1)$$

$$X \rightarrow (\text{zai } X_1 \text{ qian}, \text{before } X_1)$$

$$X \rightarrow (X_1 \text{ zhiyi}, \text{one of } X_1)$$

Summary

- Synchronous context-free grammars vary in power depending on rank
- Translation is easy; bitext parsing is exponential in rank

Summary

- Beyond synchronous CFGs,
  - synchronous TSGs allow multilevel rules
  - synchronous TAGs allow discontinuous constituents