

---

# Advanced Natural Language Processing

## Lecture 11

### Grammar Formalisms (II)

Frank Keller (slides by Mark Steedman and Bonnie Webber)

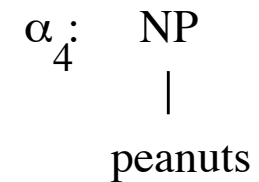
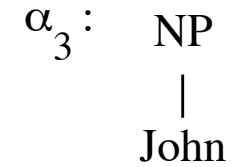
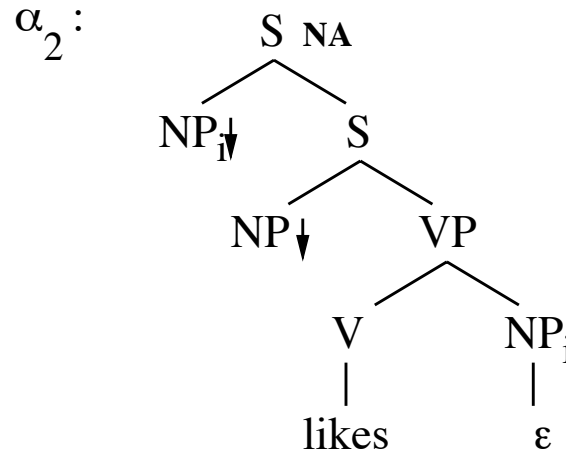
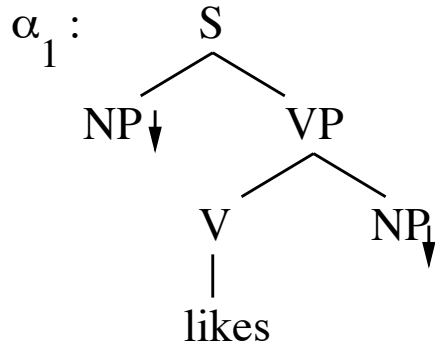
14 October 2011



## Lexicalized Tree Adjoining Grammar (LTAG)

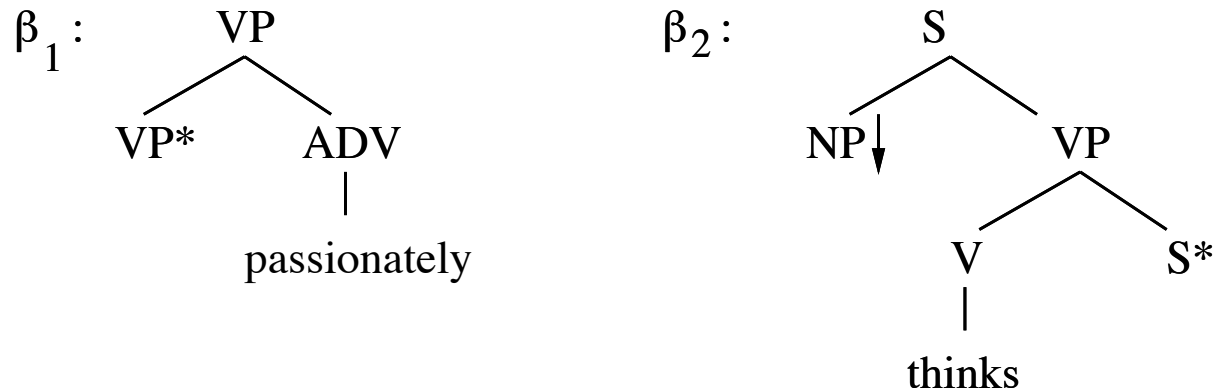
- Lexicalized Tree Adjoining Grammar (LTAG) associates an *elementary tree* with every lexical item.
- Elementary trees are either *initial trees* or *auxiliary trees*.
- They combine by two operations of *substitution* (which is just what it says) and *adjunction* (which is a little harder to explain).
- Substitution and Adjunction are reminiscent of the generalized transformations of Chomsky 1957.

## LTAG Initial ( $\alpha$ ) Trees



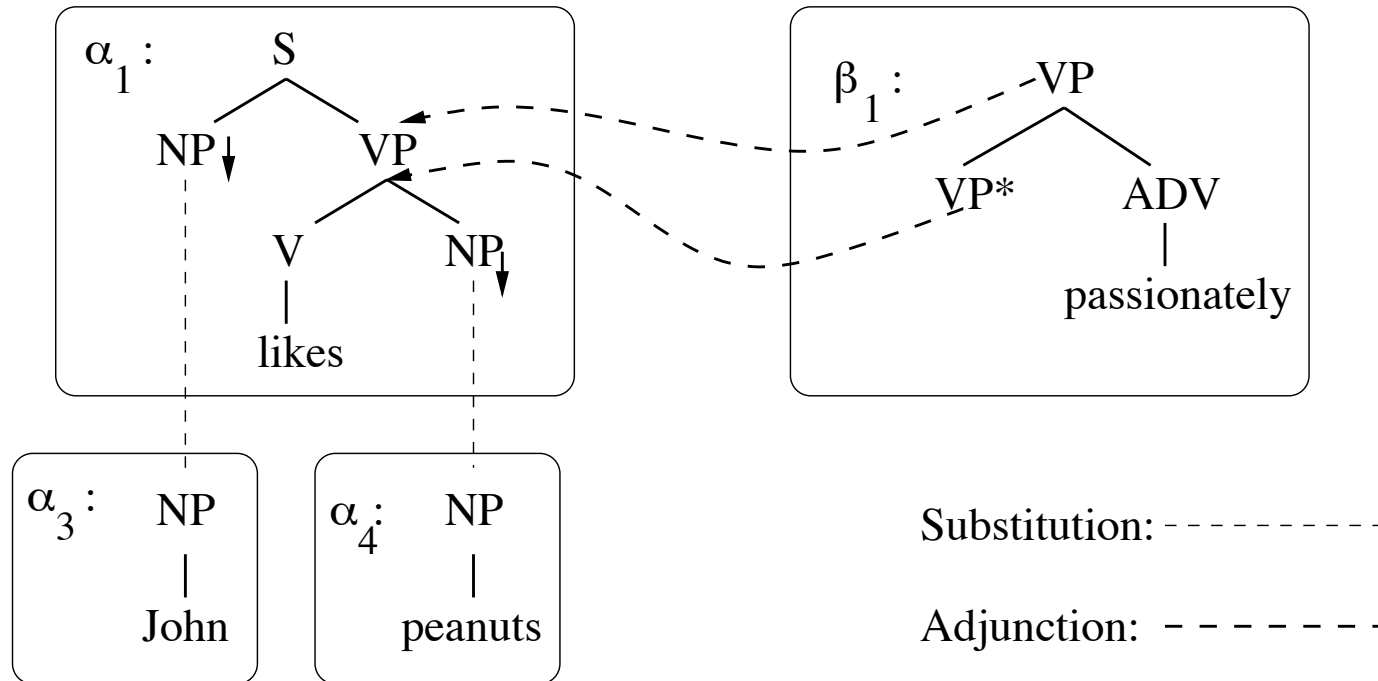
- Down arrow  $\downarrow$  on a category like NP means “an NP must be substituted here.”

## LTAG Auxiliary ( $\beta$ ) Trees



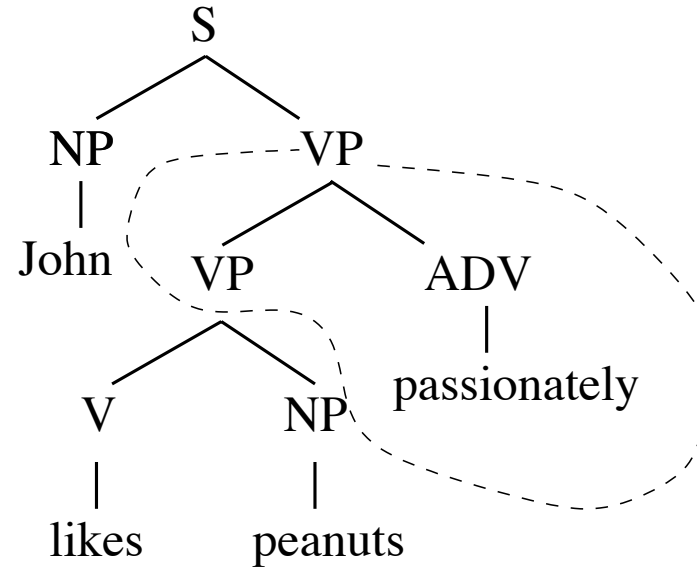
- An asterisk \* on a category like S means “this S must adjoin to S.” **NA** on a category like S means “adjoining at this S is not allowed.” The NA annotation on the initial tree in the last slide will prevent *\*Bill thinks peanuts, John likes.*

# A Derivation involving Substitution and Adjunction

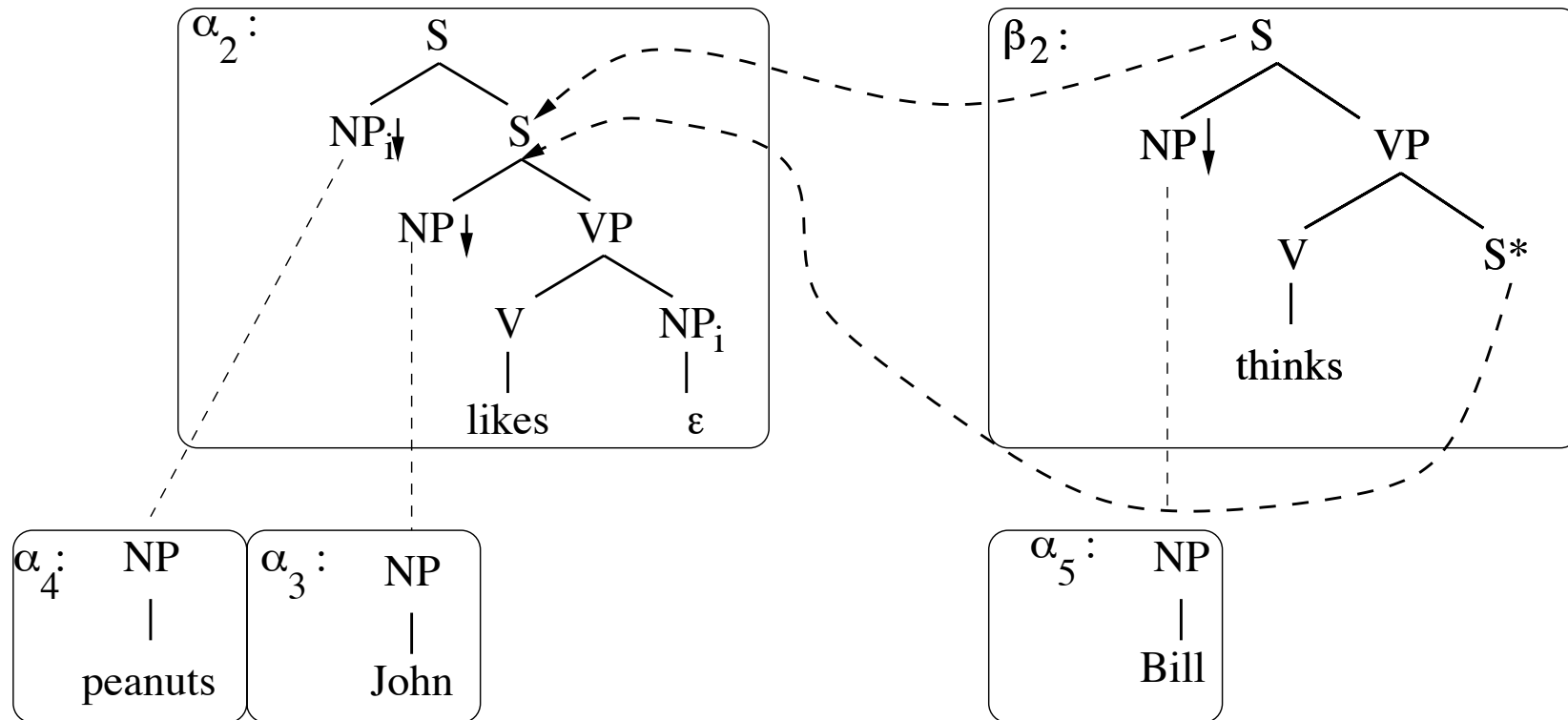


- Recall that a asterisk \* on a category like S means “this S must adjoin to S.”

## A Derivation

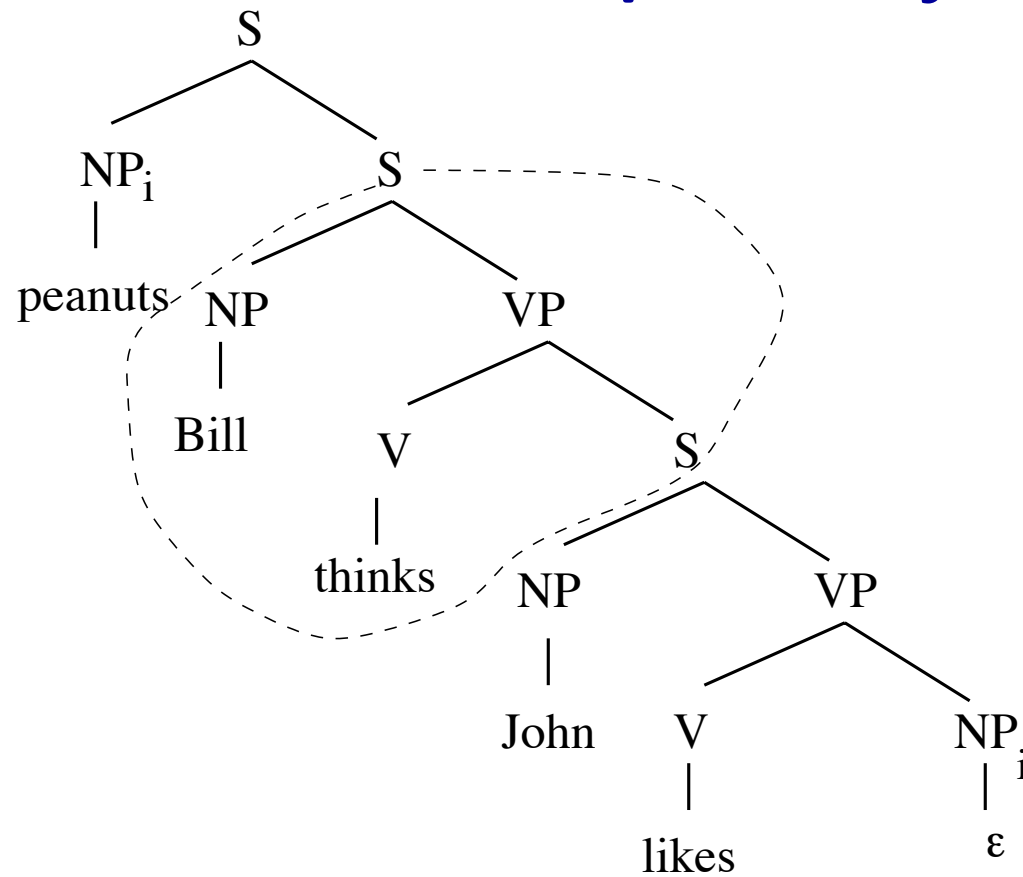


# Unbounded Dependency: Topicalization

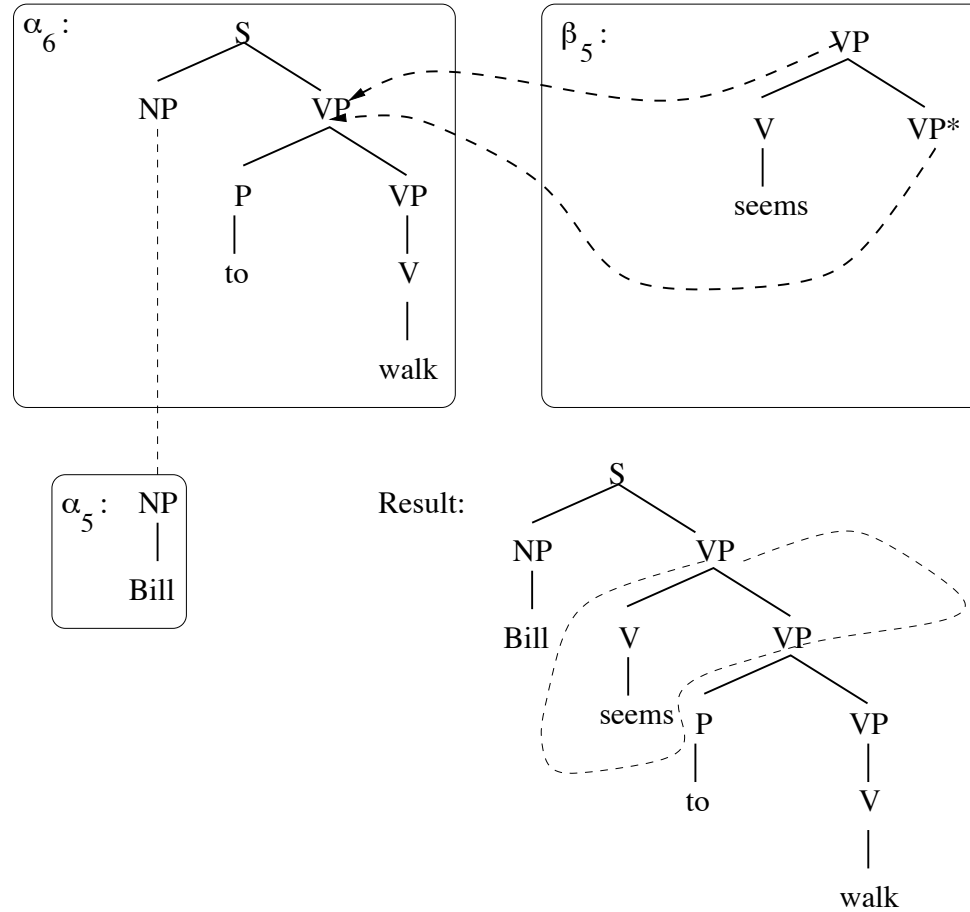


- Clearly we could adjoin unboundedly many auxiliary trees like  $\beta_2$ .

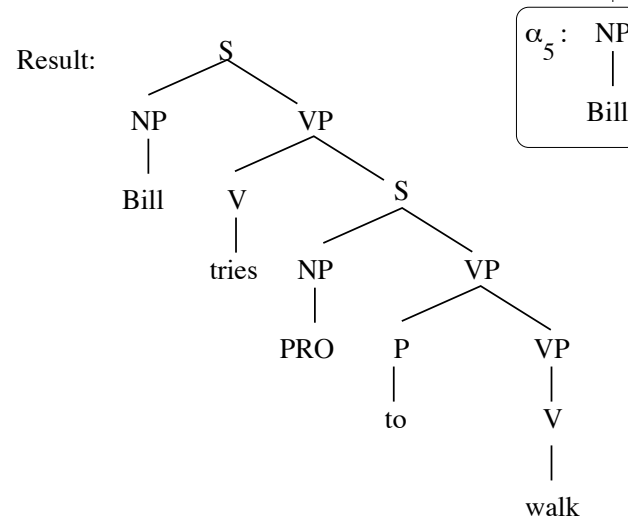
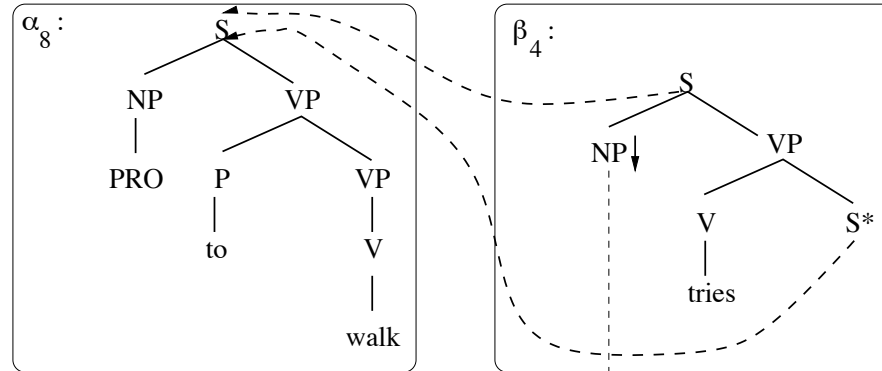
## Unbounded dependency



# The "Raising" Construction in TAG



# The “Control” Construction in TAG

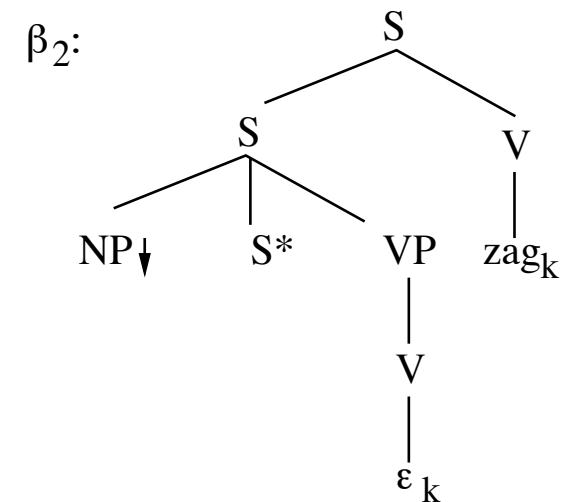
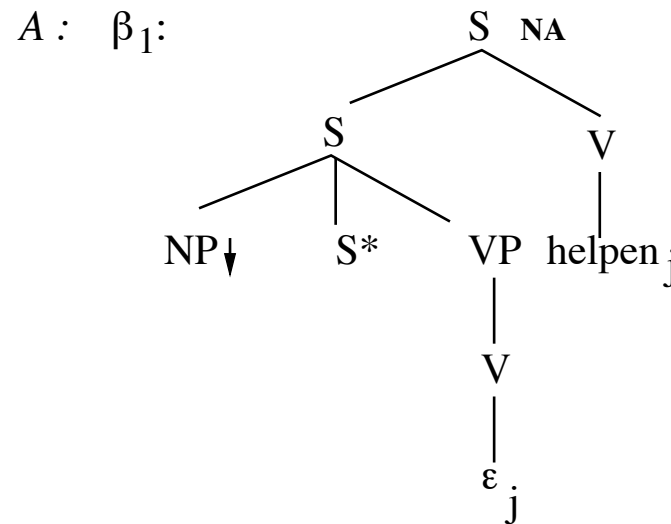
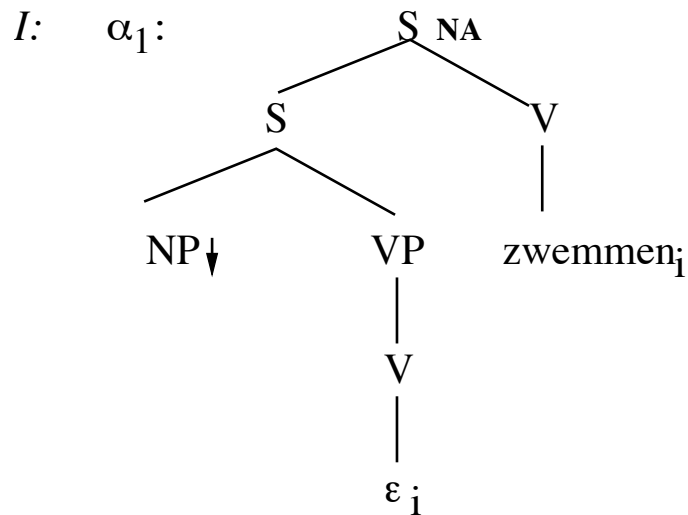


## LTAG Can be Expressed as an LIG

- LTAG lexical elementary trees like  $\alpha_1$ ,  $\alpha_2$ , and  $\beta_2$ , can be regarded as bearing a stack-valued feature, where the elements on the stack are the  $\downarrow$  and  $*$  marked leaf nodes, and where their order on the stack is defined by a leftmost-depth-first walk along the arcs of the tree.
- Substitution and adjunction can then be thought of as operations which pass a single such stack-valued feature to their result.
- This intuition underpins Weir's 1988 identification of the equivalence between LTAG and LIG [Joshi, 1991].
- Seen in that light its not too surprising that we can capture the trans-CF grammar for  $a^n b^n$  crossing.

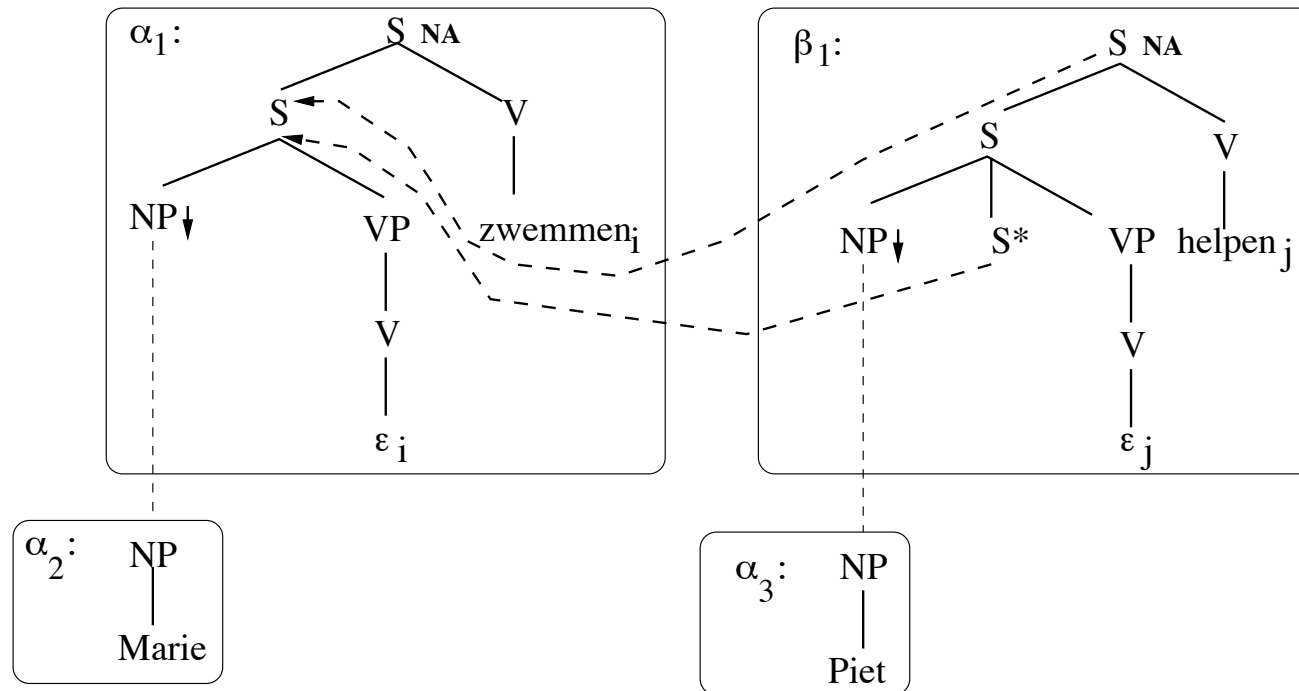
## Crossing Dependencies in LTAG ( $a^n b^n$ )

- Consider the  $G = (I, A)$  where  $I, A$  are the following sets of Initial and Auxiliary trees:



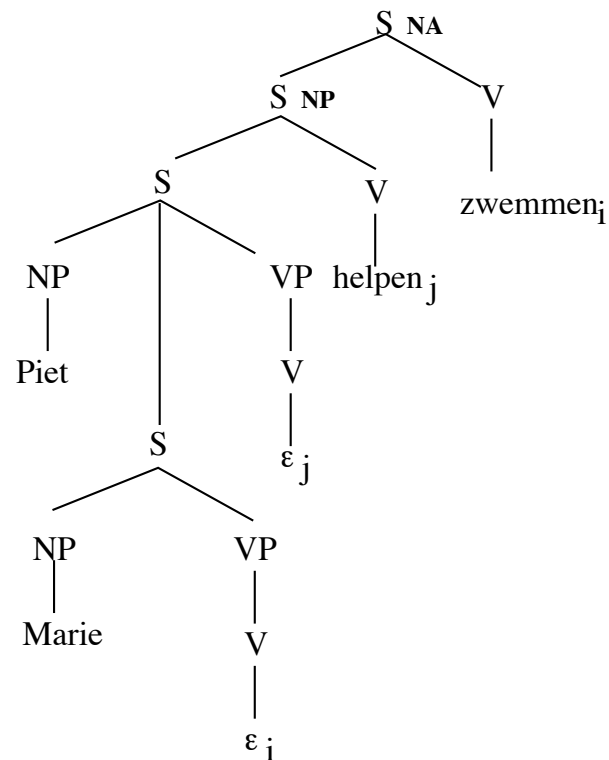
# Crossing Dependencies

- These categories engender crossing dependencies between lexical sisters in *(dat) Piet Marie helpen zwemmen*



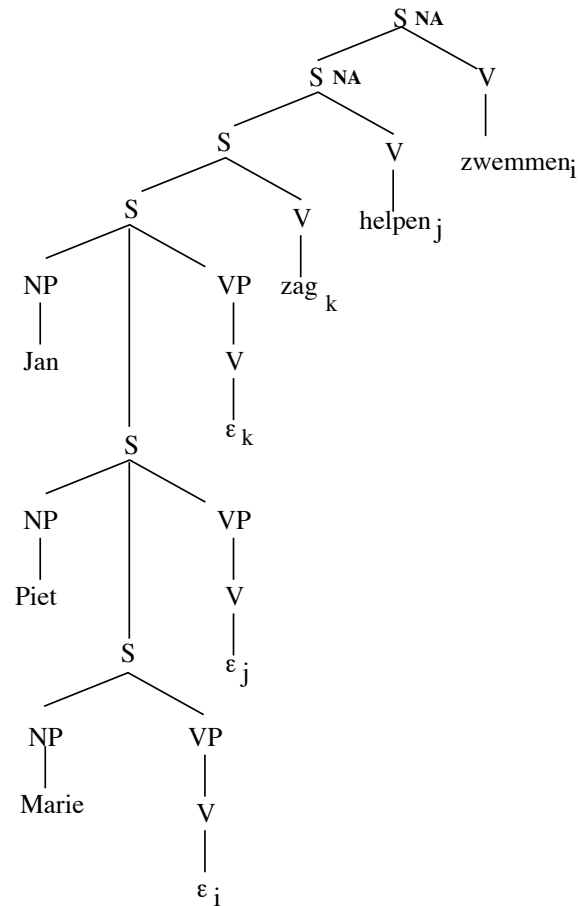
## Crossing Dependencies – One adjunction

- *(dat) Piet Marie helpen zwemmen*





# Crossing Dependencies – Two adjunctions



## Discussion

- So LTAG provides a strongly adequate account of unbounded and crossing dependencies.
- With coordination, it can account for some structures which involve crossing:
  - (dat) Jan Piet Marie zag helpen zwemmen en hoorde leren zingen.  
(that) Jan Piet Marie saw help swim and heard teach sing.  
*(that) Jan saw Piet help Marie to swim and heard Piet teach Marie to sing.*

## Discussion

- But other coordinated structures, it cannot account for:
  - (dat) Jan Piet Marie en ik Cecilia de nijlpaarden zag helpen zwemmen.  
(that) Jan Piet Marie and I Cecilia the hippos saw help swim.  
*(that) Jan saw Piet help Marie and I saw Cecilia help the hippos to swim.*
  - (dat) Jan Piet en ik Cecilia de nijlpaarden zag helpen zwemmen.  
(that) Jan Piet and I Cecilia the hippos saw help swim.  
*(that) Jan saw Piet and I saw Cecilia help the hippos to swim.*
- In Dutch hippo sentences, any contiguous sequence of functions and arguments can coordinate.