LFG

Grammatikformalismen
Sommer Semester 2010

Antske Fokkens

Department of Computational Linguistics
Saarland University

17 November 2009

Outline

Outline

Lexical Functional Grammar, Introduction

- Developed in the late 70s by Joan Bresnan and Ron Kaplan
- LFG brings scholars from different fields together:
  - Theoretical linguists
  - Descriptive, typological linguists
  - Computational linguistics
- Main ideas:
  - A formal system to model human speech (fits in the tradition of generative grammar)
  - Psychological plausibility: the formalism should be able to represent a native speaker's syntactic knowledge appropriately
  - Strong typological basis: analyses should capture cross-linguistic similarities

Main levels of representation

A Lexical Functional Grammar represents expressions in (minimally) two levels of representation:

- **constituent structure** (c-structure):
  - a tree which represents phrase structure configurations
  - it indicates the superficial arrangements of the words in the sentence, i.e. it serves as an input for the phonological interpretation of the string
  - languages differ radically on a c-structure level
- **functional structure** (f-structure):
  - an attribute-value matrix represents surface grammatical functions, i.e. traditional syntactic relations such as subject, object, complement and adjunct
  - It serves as the sole input to the semantic component
  - languages are similar on a f-structure level

Lexical Functional Grammar

- LFG is **lexical** because of the assumption that words and lexical items are as important in providing grammatical information as syntactic elements
- LFG is **functional** because grammatical information is represented by lexical functions (f-structure), rather than by phrase structure configurations

Outline

F-structure: motivation

- Assumption: for any language functional syntactic concepts such as subject and object are relevant
- The f-structure can represent what languages have in common in wide-spread phenomena, no matter how radically different languages may be on the surface

  e.g. passives
- The f-structure can capture some universal properties of language

  e.g. the Keenan-Comrie Hierarchy for relative clauses:
  SUBJ > DOBJ > IOBJ > OBL > GEN > OCOMP
  - A language may sets its border for acceptable and unacceptable relative clauses anywhere on the hierarchy: those elements above the boundary can be relativized.
  - Processing becomes more difficult when going down the hierarchy

An example of an F-structure

- Example: the f-structure of *I saw the girl*:

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'pro'} \\ \text{PERS} & 1 \\ \text{NUM} & \text{SG} \end{bmatrix} \\ \text{TENSE} & \text{PAST} \\ \text{PRED} & \text{'see}\left\langle (\uparrow\text{SUBJ}),(\uparrow\text{OBJ}) \right\rangle\text{'} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{'girl'} \\ \text{DEF} & + \\ \text{PERS} & 3 \\ \text{NUM} & \text{SG} \end{bmatrix} \end{bmatrix}$$

Formal properties of F-structures

- An F-structure is a finite set of pairs of attributes and values
- An F-structures attributes may be
  - A: atomic symbols, e.g. SUBJ, OBJ, PRED
- An F-structures values may be:
  - A: atomic symbols, e.g. SG, 1, +, PAST
  - S: semantic forms, e.g. 'girl', 'see<(↑SUBJ)(↑ OBJ)>'
  - F: f-structures
- F-structures are defined by the following recursive domain equation:

  $F = (A \rightarrow_f F \cup A \cup S)$

Examples of simple F-structures

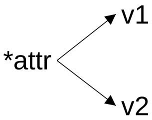
- $$f: \begin{bmatrix} \textsc{pred} & \text{'David'} \\ \textsc{num} & \textsc{sg} \end{bmatrix}$$

  Description:
  $(f\ \textsc{pred}) = \text{'David'}$
  $(f\ \textsc{num}) = \textsc{sg}$

- $$g: \begin{bmatrix} \textsc{pred} & \text{'smile}(\textsc{subj})\text{'} \\ \textsc{tense} & \textsc{past} \\ \textsc{subj} & f \begin{bmatrix} \textsc{pred} & \text{'David'} \\ \textsc{num} & \textsc{sg} \end{bmatrix} \end{bmatrix}$$

  Description:
  $(g\ \textsc{pred}) = \text{'smile}(\textsc{subj})\text{'}$
  $(g\ \textsc{tense}) = \textsc{past}$
  $(g\ \textsc{subj}) = f$

A Functional structure

- Mathematically, the f-structure can be is seen as a function from attributes to values, hence its name
- A function assigns a unique value to its argument
- In other words:
  - if (f q) = t and (f q) = v, then t = v
  - 

$v1 \neq v2$

F-structure values (additional possibilities)

- The value of an attribute can be a set:
  - $$\begin{bmatrix} \text{attr1} & \text{v1} \\ \text{attr2} & \{\text{v2,v3}\} \end{bmatrix} \text{ e.g. } \textit{we}\text{: } \begin{bmatrix} \text{PRED} & \text{'pro'} \\ \text{PERS} & \{\text{H,S}\} \\ \text{NUM} & \text{PL} \end{bmatrix}$$
- The value of an attribute can be hybrid:
  - $$\begin{bmatrix} \text{NUM PL} \\ \left\{ \begin{bmatrix} \text{PRED} & \text{'girl'} \\ \text{NUM} & \text{sg} \end{bmatrix} \begin{bmatrix} \text{PRED} & \text{'boy'} \\ \text{NUM} & \text{sg} \end{bmatrix} \right\} \end{bmatrix}$$

symbols and semantic forms

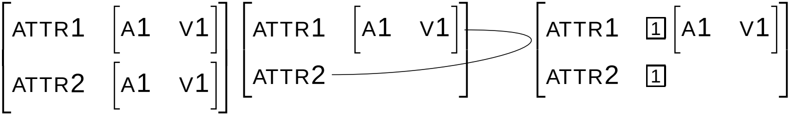
- Symbols are unbroken strings of alphanumeric characters $\rightarrow$ the choice of symbols belongs to a particular theory of linguistics
- Semantic forms are special: the single quotes around semantic form values indicate that this form is unique. E.g. each instance of the word *girl* is a uniquely instantiated occurrence of the semantic form 'girl'

Some Linguistic terminology (Bresnan 1982)

- an attribute-value pair where the value is a symbol is called a **feature**
- an attribute-value pair where the value is an f-structure is called a **grammatical function**
- an attribute whose value is a semantic form is called a **semantic feature**

Attributes with the same values

- Two attributes within the same f-structure can have the same value
- This can be represented in several ways:



- Note:
  - Semantic forms are unique: two instances of ’lion’ in a sentence does not necessarily mean two attributes have the same value: co-indexation is required

Grammatical functions in LFG

LFG proposes the following inventory of grammatical functions, which is universally available:

- SUBJect
- OBJect
- OBJ_θ
- COMP
- XCOMP
- OBLique_θ
- ADJunct
- XADJunct

Governable grammatical functions (regierbare Funktionen)

- SUBJ, OBJ, XCOMP, COMP, OBJ_θ and OBL_θ are *governed* or *subcategorized for* by the predicate, hence the name **governable grammatical functions**
- ADJ and XADJ modify the phrase they appear in, but they are not subcategorized for by the predicate. The term **modifiers** applies to these functions

The value of ADJ and XADJ

- In principle, there is no limit to the number of modifiers that can appear within a phrase: the value of the ADJ or XADJ feature is the set of all modifiers that are present, e.g. *David smiled quietly* (*yesterday*):

$$\left[\begin{array}{ll} \text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{'David'} \end{array}\right] \\ \text{PRED} & \text{'smile}<(\uparrow \text{SUBJ})>\text{'} \\ \text{ADJ} & \left\{\left[\begin{array}{ll} \text{PRED} & \text{'quietly'} \end{array}\right]\right\} \end{array}\right] \left[\begin{array}{ll} \text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{'David'} \end{array}\right] \\ \text{PRED} & \text{'smile}<(\uparrow \text{SUBJ})>\text{'} \\ \text{ADJ} & \left\{\begin{array}{l} \left[\begin{array}{ll} \text{PRED} & \text{'quietly'} \end{array}\right] \\ \left[\begin{array}{ll} \text{PRED} & \text{'yesterday'} \end{array}\right] \end{array}\right\} \end{array}\right]$$

- Typically, the values of governable functions are not sets

Subcategorization

- A semantic form may contain an argument list, next to its semantic predicate name, e.g.
  - 'smile$<$($\uparrow$ SUBJ)$>$'
  - 'see$<$($\uparrow$ SUBJ), ($\uparrow$ OBJ)$>$'
  - 'give$<$($\uparrow$ SUBJ), ($\uparrow$ OBJ), ($\uparrow$ OBJ2)$>$'
- Note that lexical items select for grammatical functions (not for NPs, CP, etc)
- How to make sure that subcategorization requirements are fulfilled?
  $\rightarrow$ well-formedness constraints on the f-structure: completeness and coherence

Principle of completeness

- The principle of completeness requires that all governable functions present in the argument list of a semantic form must be present in the f-structure
- This excludes ungrammatical expressions such as

  * He devoured

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'pro'} \\ \text{PERS} & 3 \\ \text{NUM} & \text{SG} \end{bmatrix} \\ \text{pred} & \text{'devour}<(\uparrow\text{SUBJ}),(\uparrow\text{OBJ})>\text{'} \end{bmatrix}$$

$\longrightarrow$ the object is missing: incomplete f-structure!

Principle of Completeness: definition

Local Completeness

An f-structure is **locally complete** iff it contains all the governable functions that its predicate governs

Completeness

An f-structure is **complete** iff it is locally complete and all its subsidiary f-structures are locally complete

Principle of Coherence

- The principle of coherence requires that all governable functions present in the f-structure are also present in the argument list of the predicate
- This excludes ungrammatical examples such as
  - * David smiled the flower

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'David'} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{'flower'} \\ \text{NUM} & \text{SG} \end{bmatrix} \\ \text{PRED} & \text{'smile}<(\uparrow \text{SUBJ})>\text{'} \end{bmatrix}$$

$\rightarrow$ the OBJ *the flower* is not governed by the predicate: incoherent f-structure!

Principle of Coherence: definition

Local Coherence

An f-structure is **locally coherent** iff all the governable functions it contains are governed by its predicate

Coherence

An f-structure is **coherent** iff it is locally coherent and all its subsidiary f-structures are locally coherent

Principle of Consistency (uniqueness)

- The principle of consistency states what we have already seen in the f-structures formal properties: an attribute has a unique value
- It excludes ungrammatical examples such as

  * David smile

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'David'} \\ \text{NUM} & \text{SG/PL} \end{bmatrix} \\ \text{PRED} & \text{'smile}<(\uparrow \text{SUBJ})>\text{'} \end{bmatrix}$$

  $\longrightarrow$ 'David' is singular, but the verb form states that the subject's number is plural: inconsistent f-structure!

definition: An f-structure is consistent iff all attributes have at most one value

F-structures, recap I

- F-structures represent the grammatical relations of expressions
- Languages are similar on this level: allows to explain cross-linguistic properties of phenomena
- Formally, an f-structure is a set of attribute-value pairs
- LFG posits a universal inventory of grammatical functions (where we distinguish governable functions and modifiers (among other properties))
- F-structures must be
  - complete
  - coherent
  - consistent

Outline

Constituent structure

- The constituent structure represents the organization of overt phrasal syntax
- It provides the basis for phonological interpretation
- Languages are very different on the c-structure level

Constituency

- Why constituency?
  - Example *the dachshund is barking*

  → Observations by Noam Chomsky:
    - The same sequence of categories may appear in more than one environment e.g. *David petted the dachshund*
    - Such sequences can be replaced by the same sequence with additional modifiers *the black dachshund is barking*, *David petted the black dachshund*

    → constituents capture the intuitions that certain sequences form phrasal units (e.g. *the dachshund*), and others do not (e.g. *petted the*)

    → constituents simplify linguistic description: distribution can be defined for a phrase, and need not be defined for each individual sequence of words
- What is a constituent?

How to identify constituents?

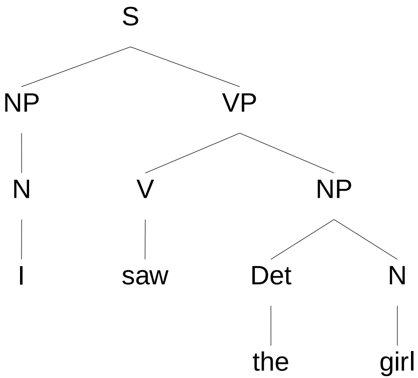
There are several tests to identify constituents:

- Distribution: can the sequence occur in a variety of other sentence positions?
- Questions: is the sequence an answer to *who*, *what*, *how*, *where*?
- Scrambling: can the sequence be topicalized? Appear in the first position of a verb-second language?
- Non-separability: are there elements that may not be inserted in the sequence?

Properties of c-structures

- C-structures are conventional phrase structure trees: they are defined in terms of syntactic categories, terminal nodes, dominance and precedence
- They are determined by a context free grammar that describes all possible surface strings of the language

Example of a c-structure

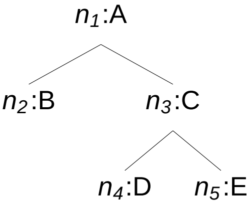


Properties of a tree (Kaplan 1995)

- A tree consists of:
  - N: a set of nodes
  - M: N $\rightarrow$ N
    a mother function M that takes nodes into nodes
  - $< \subseteq$ N x N
    a partial ordering $<$
  - λ: N $\rightarrow$ L
    Nodes are related by a labeling function λ that takes nodes into some finite labeling set L
- LFG admits only nontangled trees:
  - For any nodes n_1 and n_2, if M(n_1) $<$ M(n_2), then $n_1 < n_2$

Description of a tree

- Tree:



- Description of the tree:

$$\begin{array}{lll}
M(n_2) = n_1 & \lambda(n_1) = A & \lambda(n_2) = B \\
M(n_3) = n_1 & \lambda(n_3) = C & n_2 < n_3 \\
M(n_4) = n_3 & \lambda(n_4) = D & M(n_5) = n_3 \\
\lambda(n_5) = E & n_4 < n_5 &
\end{array}$$

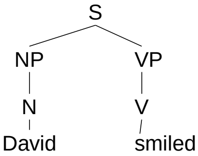
Outline

structural correspondences

- C-structures and f-structures represent different properties of an utterance
- How can these structures be associated properly to a particular sentence?
- Words and their ordering carry information about the linguistic dependencies in the sentence
- This is represented by the c-structure (licensed by a CFG)
- LFG proposes simple mechanisms that maps between elements from one structure and those of another: correspondence functions
- A function ϕ allows to map c-structures to f-structures
  ϕ: N $\rightarrow$ F

Mapping from c- to f-structure: The head convention

- Consider the following example:

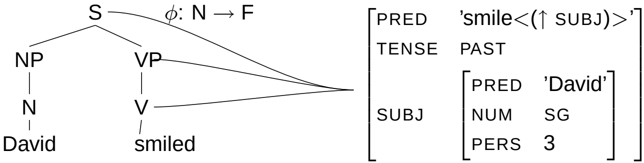


$$\begin{bmatrix} \text{PRED} & \text{'smile}<(\uparrow \text{SUBJ})>\text{'} \\ \text{TENSE} & \text{PAST} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'David'} \\ \text{NUM} & \text{SG} \\ \text{PERS} & 3 \end{bmatrix} \end{bmatrix}$$

- The **head convention** states that a phrase inherits its functional properties and requirements from its head: a constituent structure phrase and its head map to the same f-structure
- S, VP and V thus map to the same f-structure

Mapping from c- to f-structure: The head convention

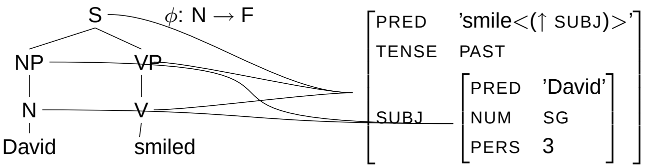
- Consider the following example:



- The **head convention** states that a phrase inherits its functional properties and requirements from its head: a constituent structure phrase and its head map to the same f-structure
- S, VP and V thus map to the same f-structure

Mapping from c- to f-structure: The head convention

- Consider the following example:



- The **head convention** states that a phrase inherits its functional properties and requirements from its head: a constituent structure phrase and its head map to the same f-structure
- S, VP and V thus map to the same f-structure

Annotating PS-rules: heads

- Consider the following rule to expand VP to V

  $$VP \longrightarrow V$$

- We express the fact that VP and V have the same f-structure by annotating the V-node:

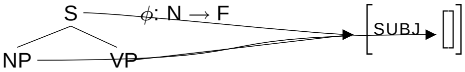
  $$VP \longrightarrow \underset{\phi(M(n)) = \phi(n)}{V}$$

- This equation indicates that the f-structure of the mothernode of V ($\phi(M(n))$) is equal to the node of V ($\phi(n)$)
  - An alternative notation:

    $$VP \longrightarrow \underset{\uparrow = \downarrow}{V}$$

Annotating PS-rules: grammatical functions

- Consider the following example:



- Here the NP bears the SUBJ function
- The following phrase structure rule carries the additional information to derive the correct f-structure:

$$S \rightarrow \underset{(\phi(M(n))\ \text{SUBJ}) = \phi(n)}{NP} \quad \underset{\phi(M(n)) = \phi(n)}{VP}$$

- An alternative notation:

$$S \rightarrow \underset{(\uparrow\ \text{SUBJ}) = \downarrow}{NP} \quad \underset{\uparrow = \downarrow}{VP}$$

Lexical Entries

In lexical entries, information about the item's f-structure is represented in the same way as in c-structures:

smiled V ($\uparrow$ PRED) = 'smile<($\uparrow$ SUBJ)>'
($\uparrow$ TENSE) = PAST

- The equivalent phrase structure rule:

V $\rightarrow$ *smiled*
($\uparrow$ PRED) = 'smile<($\uparrow$ SUBJ)>'
($\uparrow$ TENSE) = PAST

An example analysis: *David smiled*

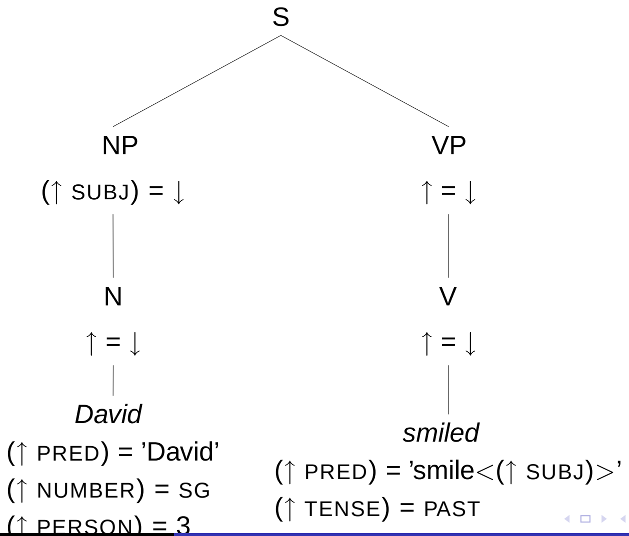
- We assume the following annotated PS-rules:
  - S $\rightarrow$ NP VP
    $(\uparrow \text{SUBJ}) = \downarrow$ $\uparrow = \downarrow$

    VP $\rightarrow$ V
    $\uparrow = \downarrow$

    NP $\rightarrow$ N
    $\uparrow = \downarrow$
- and the following lexical entries
  - *smiled* V $(\uparrow \text{PRED})$ = 'smile$<(\uparrow \text{SUBJ})>$'
    $(\uparrow \text{TENSE}) = \text{PAST}$

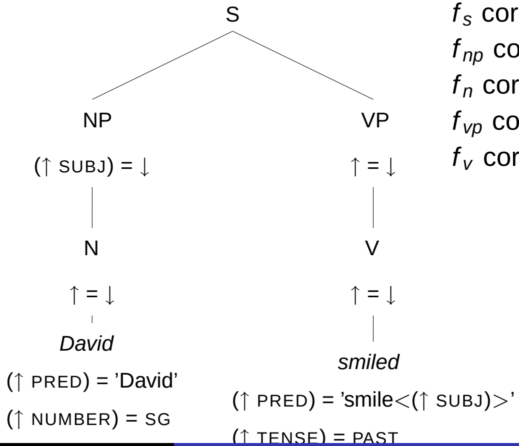
    David N $(\uparrow \text{PRED})$ 'David'
    $(\uparrow \text{NUMBER}) = \text{SG}$
    $(\uparrow \text{PERSON}) = 3$

Analysis of *David smiled*



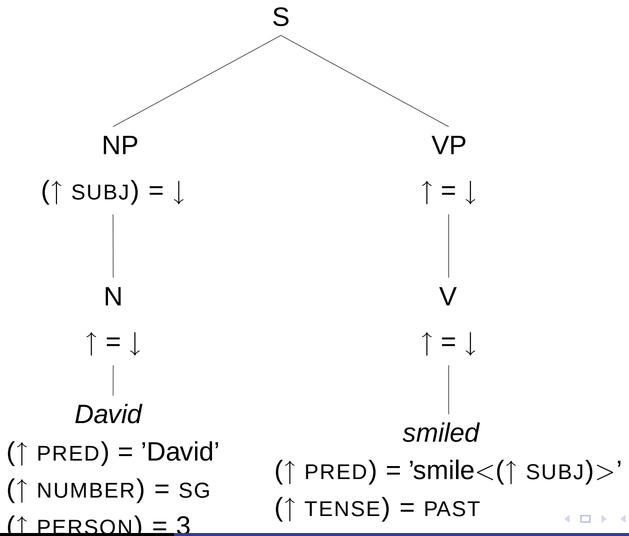
Instantiating the f-description of the sentence

- In order to get the functional description of the sentence, we associate each node with an f-structure:

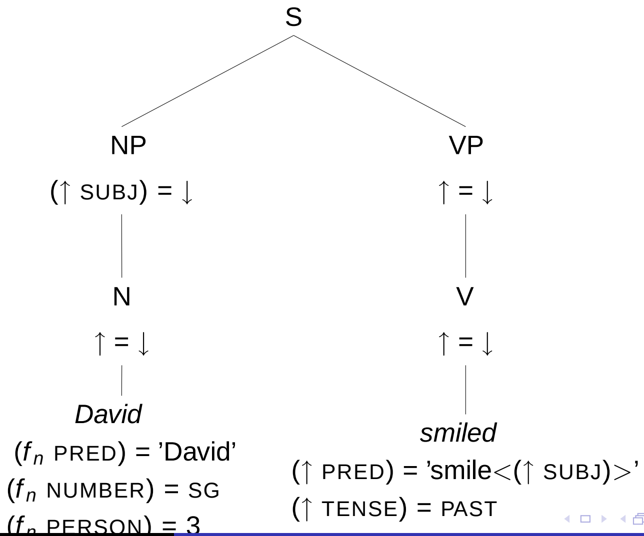


f_s corresponds to node S
f_{np} corresponds to node NP
f_n corresponds to node N
f_{vp} corresponds to node VP
f_v corresponds to node V

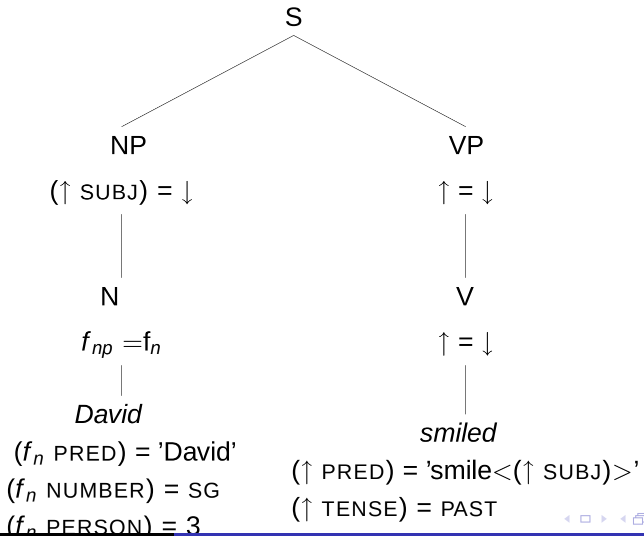
References of $\uparrow$ and $\downarrow$



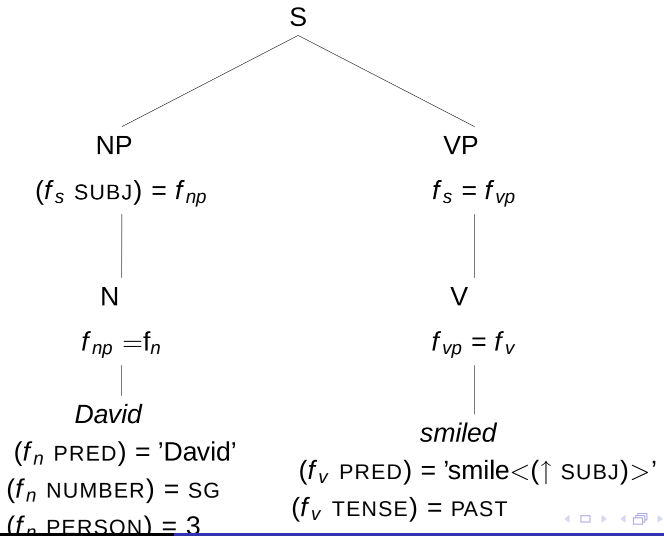
References of ↑ and ↓



References of ↑ and ↓



References of ↑ and ↓



The functional description

- The tree on the previous slide provides the following functional description:

$(f_s \text{ SUBJ}) = f_{np}$
$f_{np} = f_n$
$(f_n \text{ PRED}) = \text{'David'}$
$(f_n \text{ NUMBER}) = \text{SG}$
$(f_n \text{ PERSON}) = 3$
$f_s = f_{vp}$
$f_{vp} = f_v$
$(f_v \text{ PRED}) = \text{'smile} < (\uparrow \text{SUBJ}) > \text{'}$
$(f_v \text{ TENSE}) = \text{PAST}$

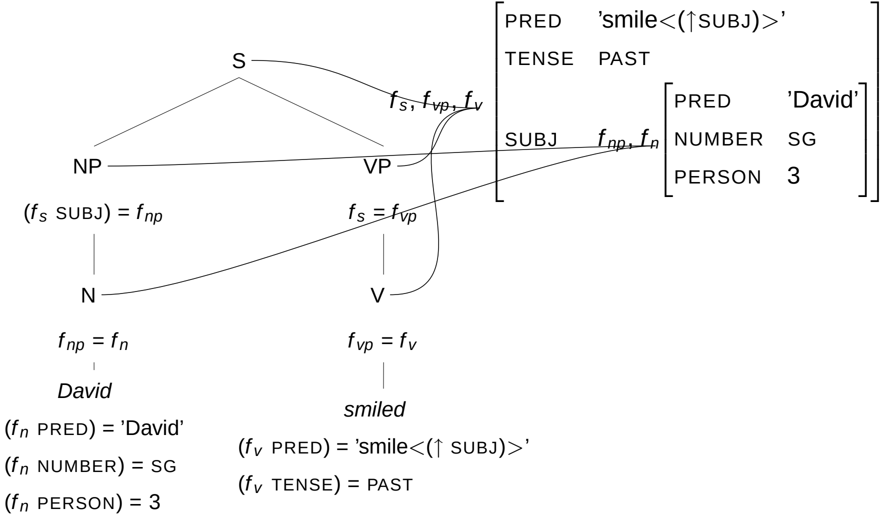
The functional description

- The tree on the previous slide provides the following functional description:

$(f_s \text{ SUBJ}) = f_{np}$
$f_{np} = f_n$
$(f_n \text{ PRED}) = \text{'David'}$
$(f_n \text{ NUMBER}) = \text{SG}$
$(f_n \text{ PERSON}) = 3$
$f_s = f_{vp}$
$f_{vp} = f_v$
$(f_v \text{ PRED}) = \text{'smile}<(\uparrow \text{SUBJ})>\text{'}$
$(f_v \text{ TENSE}) = \text{PAST}$

$$f_s, f_{vp}, f_v \begin{bmatrix} \text{PRED} & \text{'smile}<(\uparrow\text{SUBJ})>\text{'} \\ \text{TENSE} & \text{PAST} \\ \text{SUBJ} & f_{np}, f_n \begin{bmatrix} \text{PRED} & \text{'David'} \\ \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \end{bmatrix}$$

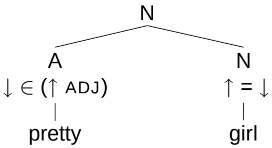
David smiled: f- and annotated c-structure



Adjuncts

- The attribute ADJ takes a set as its value
- The c-structure/f-structure correspondance rule expresses membership to a set as follows:

$$\text{N} \rightarrow \underset{\downarrow \in (\uparrow \text{ ADJ})}{\text{AdjP}} \qquad \underset{\uparrow = \downarrow}{\text{N}}$$

- 

$$\begin{bmatrix} \text{PRED} & \text{'girl'} \\ \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \\ \text{ADJ} & \left\{ \begin{bmatrix} \text{PRED} & \text{'pretty'} \end{bmatrix} \right\} \end{bmatrix}$$

Bibliography I

- Bresnan, Joan (2000). *Lexical Functional Syntax*. Blackwell Publishers: Malden, USA/Oxford UK.
- Dalrymple, Mary, Ron M. Kaplan, John T. Maxwell III and Annie Zaenen (eds.). (1995) *Formal Issues in Lexical-Functional Grammar*. CSLI Publications: Palo Alto, USA.
- Dalrymple, Mary (2001). *Lexical Functional Grammar*. Academic Press: San Diego, USA/London, UK.
- Kaplan, Ron (1995). The formal architecture of Lexical-Functional Grammar. In: Dalrymple et al. (1995).
- Kordoni, Valia (2008a). Syntactic Theory Lectures 5. Course slides.
- Schneider, Gerold (1998). *A Linguistic Comparison of Constituency, Dependency and Link Grammar. Lizentiatsarbeit, Institut für Informatik der Universität Zürich*. http://www.ifi.unizh.ch/cl/study/lizarbeiten/lizgerold.pdf.