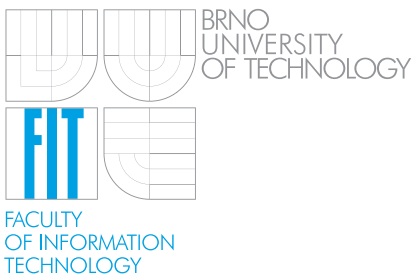
Lexical Function Grammar

Petr Horáček, Eva Zámečníková and Ivana Burgetová

Department of Information Systems
Faculty of Information Technology
Brno University of Technology
Božetěchova 2, 612 00 Brno, CZ



FRVŠ MŠMT FR97/2011/G1

Outline

- Introduction
- F-structures
- Constraints of f-structures
- C-structures
- Mapping c-structure to f-structure

- **Introduction**
- F-structures
- Constraints of f-structures
- C-structures
- Mapping c-structure to f-structure

What is Lexical Functional Grammar?

Motivation

1. Lexical – to have richly structured lexicon, where relations between eg. verbal alternations are stated.

2. Functional – abstract grammatical functions like subject and object are primitives.

What is Lexical Functional Grammar?

LFG is a theory of:

- Syntax – how words can be combined together to make larger phrases, such as sentences.
- Morphology – how morphemes can be combined to make up words.
- Semantics – how and why various words and combinations of words mean what they mean
- Pragmatics – how expressions are used to transmit information.

- *morphemes* = parts of words, eg. writers, namely the verb write, the 'agentive affix' er and the plural marker +s
- Grammar is often taken to include phonology (the study of the sound systems of human languages).

Lexical Functional Grammar

- LFG consists of multiple dimensions of structure.
- Each of these dimensions is represented as a distinct structure with its own rules, concepts, and form.

LFG minimally distinguishes two kinds of representations:

- c-structure – the structure of syntactic constituents.
- f-structure – the representation of grammatical functions.

These are two completely different formalisms:

- *trees* for c-structure.
- *attribute-value matrices* for c-structure.

Other types of structures in LFG

There are also other kinds of structures:

- argument structure
- semantic structure
- information structure
- morphological structure
- phonological structure

The various structures can be said to be mutually constraining.

- Introduction
- **F-structures**
- Constraints of f-structures
- C-structures
- Mapping c-structure to f-structure

F-structures

- F-structures maps closely to meaning and
- encodes abstract grammatical relations like subject and object as *primitives*, i.e. they are not reducible to anything else.

- Categories like subject and object are *cross-linguistic* → languages vary less in their *f-structure*

F-structures: Grammatical functions

Example

We have this inventory:
SUBJect, OBJect, OBJ_θ, COMP, XCOMP, OBLique_θ, ADJunct, XADJunct

- Terms (core functions): SUBJ, OBJ, OBJ_θ
- Semantically restricted:
  - OBJ_θ: secondary OBJ function associated with thematic roles ($\text{OBJ}_{\text{THEME}}$)
  - OBL_θ: thematically restricted oblique functions
- Open clausal functions:
  - COMP: sentencial or closed infinitival complement
  - XCOMP: open (predicative) complement with externally controlled subject

Subcategorization

- Verbs select for gramatical functions
- Use the predicate feature PRED to specify the semantic form:

- *yawn*: PRED 'YAWN<SUBJ>'
- *hit*: PRED 'HIT<SUBJ, OBJ>'
- *give*: PRED 'GIVE<SUBJ, OBJ, $\text{OBJ}_{\text{THEME}}$ >'
- *eat*: PRED 'EAT<SUBJ, (OBJ)>'

Simple f-structures

F-structure is a function from attributes to values.

Example

For the noun **David**:

- PRED and NUM are attributes.
- DAVID and SG are the corresponding values.

$$\begin{bmatrix} PRED & 'DAVID' \\ NUM & SG \end{bmatrix}$$

Example

F-structures within f-structures: ***David yawned.***

$$\begin{bmatrix} PRED & 'YAWN < SUBJ >' \\ TENSE & PAST \\ SUBJ & \begin{bmatrix} PRED & 'DAVID' \\ NUM & SG \end{bmatrix} \end{bmatrix}$$

Sets

Values can be sets, in order to handle phenomena with an unbounded number of elements.

Example

David yawned quietly yesterday.

$$\begin{bmatrix} PRED & 'yawn < SUBJ >' \\ TENSE & PAST \\ SUBJ & \begin{bmatrix} PRED & 'David' \\ NUM & SG \end{bmatrix} \\ ADJ & \left\{ \begin{bmatrix} PRED & 'quietly' \end{bmatrix} \right. \\ & \left. \begin{bmatrix} PRED & 'yesterday' \end{bmatrix} \right\} \end{bmatrix}$$

Sets: F-structure Representations

- *Sets* can also have additional properties = have attributes and values which apply over whole set – hybrid objects.
- Properties can distribute over elements of the set.

Example

David and Chris yawned.

$$\begin{bmatrix} \mathit{PRED} & \mathit{'yawn < SUBJ >'} \\ \mathit{TENSE} & \mathit{PAST} \\ \mathit{SUBJ} & \left\{ \begin{bmatrix} \mathit{PRED} & \mathit{'David'} \\ \mathit{NUM} & \mathit{SG} \end{bmatrix} \begin{bmatrix} \mathit{PRED} & \mathit{'Chris'} \\ \mathit{NUM} & \mathit{SG} \end{bmatrix} \right\} \end{bmatrix}$$

Attributes with Common Values

- *Attributes* can share the same values, to describe phenomena such as raising, notated in different ways.

Example

David seemed to yawn.

$$\begin{bmatrix} PRED & 'SEEM < XCOMP > SUBJ' \\ TENSE & PAST \\ SUBJ & \boxed{1}\begin{bmatrix} PRED & 'David' \\ NUM & SG \end{bmatrix} \\ XCOMP & \begin{bmatrix} PRED & 'YAWN < SUBJ >' \\ SUBJ & \boxed{1} \end{bmatrix} \end{bmatrix}$$

This is like HPSG notation.

Nature of f-structures

An f-structure is rectricted by the principles of:

1. Completeness
2. Coherence
3. Uniqueness

Definition: Completeness

- An f-structure is locally complete iff it contains all the governable grammatical functions that its predicate governs.
- An f-structure is complete iff it and all its subsidiary f-structures are locally complete.

Completeness

- List of governable grammatical functions = argument list of semantic form.
- All governable grammatical functions mentioned in the predicate must be present in the f-structure.

Example

Completeness example:

- PRED ‘DEVOUR<SUBJ, OBJ>’
- *David devoured.*

Nature of f-structures

An f-structure is rectricted by the principles of:

1. Completeness
2. Coherence
3. Uniqueness

Definition: Coherence

- An f-structure is locally coherent iff all the governable grammatical functions that it contains are governed by a local predicate.
- An f-structure is coherent iff it and all its subsidiary f-structures are locally coherent.

Example

David yawned the sink.

$$\begin{bmatrix} PRED & 'YAWN < SUBJ >' \\ SUBJ & [PRED \quad 'DAVID'] \\ XCOMP & [PRED \quad 'SINK'] \end{bmatrix}$$

An f-structure is rectricted by the principles of:

1. Completeness
2. Coherence
3. Uniqueness

Definition: Uniqueness

- In a given f-structure, a particular attribute may have at most one value.

Example

The boys yawned.

$$\begin{bmatrix} PRED & 'YAWN < SUBJ >' \\ SUBJ & \begin{bmatrix} PRED & 'BOYS' \\ NUM & SG/PL \end{bmatrix} \end{bmatrix}$$

- Introduction
- F-structures
- **Constraints of f-structures**
- C-structures
- Mapping c-structure to f-structure

Functional equations

We use functional equations on words and phrases to describe acceptable f-structures.

Example

F-description with a single equation:

$$(g\,NUM) = SG$$

Different f-structures which satisfy this f-description:

1. $$\begin{bmatrix} NUM & SG \end{bmatrix}$$

2. $$\begin{bmatrix} PRED & 'BOYS' \\ NUM & SG/PL \end{bmatrix}$$

Functional Constraints – Definition

The f-structure for an utterance is the *minimal solution* satisfying the constraints introduces by the words and phrase structure of the utterance.
Minimal solution satisfies all constraints in the f-description and has no additional structure.

Constraining Equations

- used for checking the properties of the minimal solution
- eg. the SUBJ of f must meet certain conditions:
  (f SUBJ NUM) $=_c$ *SG*

Functional Constraints – Example

Example

Lexical constraints:

- *John*
  - (g PRED) = 'JOHN'
  - (g NUM) = SG
- *runs*
  - (f PRED) = 'RUN<SUBJ>'
  - (f SUBJ CASE) = NOM
  - (f SUBJ NUM) = SG

Phrasal constraints:

- (f SUBJ) $= g$

By combining lexical and phrasal constraints we get:

- (f SUBJ) $= g$
- (g PRED) = 'JOHN'
- (g NUM) = SG
- (f PRED) = 'RUN<SUBJ>'
- (f SUBJ CASE) = NOM
- (g NUM) = SG

Example

Minimal solution:

$$\text{f:} \begin{bmatrix} PRED & 'RUN < SUBJ >' \\ SUBJ & g: \begin{bmatrix} PRED & 'JOHN' \\ CASE & NOM \\ NUM & SG \end{bmatrix} \end{bmatrix}$$

More Functional Constraints

Disjunction

Different options can be used to satisfy an f-description.

Example

I met/have met him.
Lexical entry for *met*:

- (f PRED) = 'MEET<SUBJ,OBJ>'
- {(f TENSE) = PAST|(f FORM) = PASTPART}

More Functional Constraints

Negation

It is specified what can not be true in an f-description.

Example

1. *I know whether/if David yawned.*
2. *You have to justify whether/*if your journey is really necessary.*

☹ *if* is not allowed with *justify* (*know*)

- *justify* $\vee$(*f* COMP COMPFORM) $\neq$ *IF*

Existential Constraints

An f-structure must have some attributes, but the value of that attribute is unconstrained.

Example

1. *The man who yawns/yawned/will yawn.*
2. ☹*The man who yawning.*

$\Rightarrow$ In a relative clause ,*yawn* must be tensed, but it is not important which tense.

- Relative clause constraints is: (f TENSE).
- We can also specify negative existencial constraints, e.g. $\neg$(f TENSE)

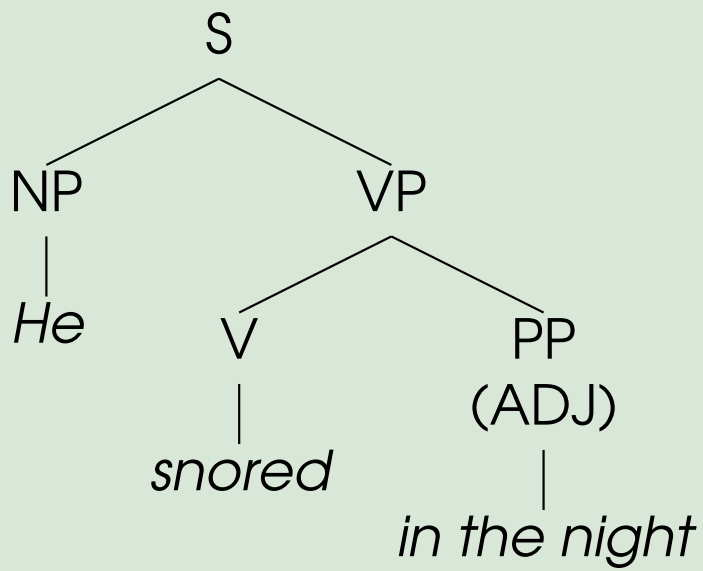
- Introduction
- F-structures
- Constraints of f-structures
- **C-structures**
- Mapping c-structure to f-structure

C-structure

- c-structure corresponds to traditional notion of *phrase grammars*.

Example

Example of a c-structure:



C-structure Rules

- c-structure rules are like phrase structure rules with a few differencies
- phrase structure rules with *optionality*, *disjunction* and *Kleene star*

We can also use:

- Metacategories
- ID/LP rules

Metarules and ID/PL Rules

Metacategories

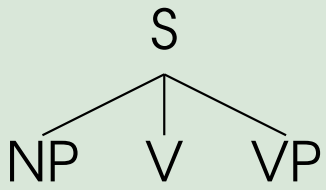
represent several different sets of categories

a. $X \equiv \{NP|PP|VP|AP|AdvP\}$

b. $VP \equiv VNP$

Example

using *VP* in b. in rule $S \rightarrow NPVP$ results in tree:



ID/PL Rules

rules can be written in ID/LP format: ID = *immediate dominance*, LP = *linear precedence*

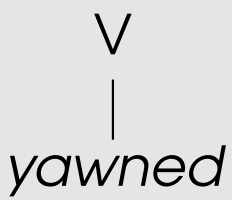
- No LP rules: $VP \rightarrow V, NP; VP \rightarrow \{V\ NP|NP\ V\}$
- One LP rule: $VP \rightarrow V, NP; VP \rightarrow V\ NP; V < NP$
- Interacting LP rules:
  $VP \rightarrow V, NP, PP; VP \rightarrow \{V\ NP\ PP|V\ PP\ NP\}; V < NP, V < PP$

C-structure: String

How a string is licensed

- context-free c-structure grammarlicenses the c-structure of a *string*
- the grammar is augmented with functional descriptions, which *map* the c-structure to an f-structure; ϕ is the *mapping function*

- Each c-structure is related to *only one* f-structure.



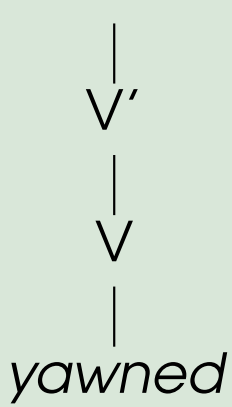
$$\phi(V): \begin{bmatrix} PRED & 'YAWN < SUBJ >' \\ TENSE & PAST \end{bmatrix}$$

- Introduction
- F-structures
- Constraints of f-structures
- C-structures
- **Mapping c-structure to f-structure**

Head Convention

- Multiple c-structures can map *onto the same* f-structure.
- This allows nodes to inherit properties from their head.

Example



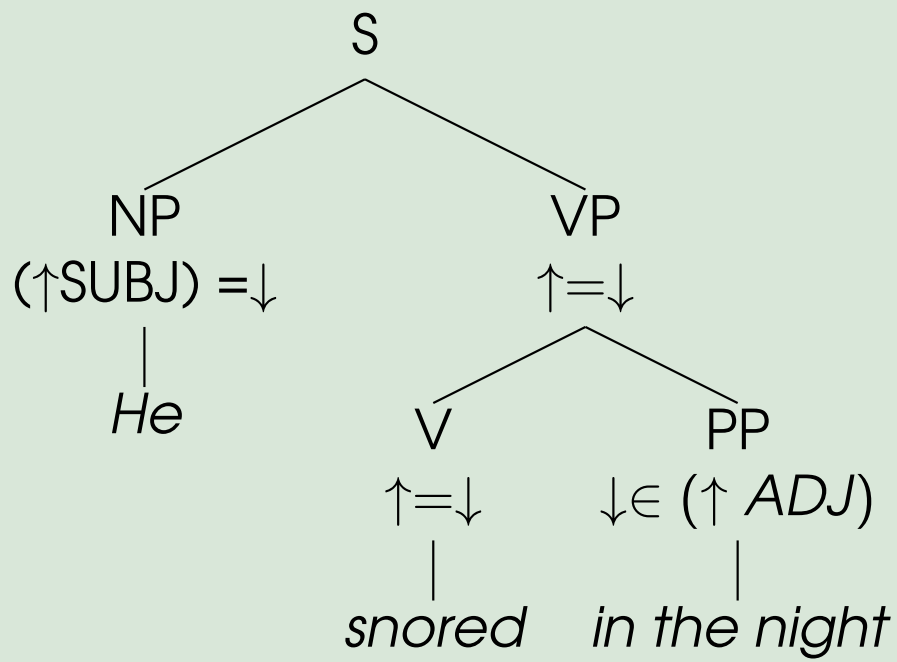
$$\phi(VP) = \phi(V') = \phi(V) : \begin{bmatrix} PRED & 'YAWN < SUBJ >' \\ TENSE & PAST \end{bmatrix}$$

Mapping c-structure to f-structure

- Functional designator $\uparrow$ refers to f-structure associated with node.
- Functional designator $\downarrow$ refers to a node's own f-structure.

Example

- $\uparrow=\downarrow$: Identifies a node's f-structure of its parent.
- $(\uparrow, \mathit{SUBJ}) = \downarrow$: Identifies a node's f-structure with the SUBJ path of it's parent's f-structure



References

Doug Arnold:
Lexical Functional Grammar (online),
Dept of Language and Linguistics, University of Essex, 2011
[cit. 2011-12-29].
http://www.essex.ac.uk/linguistics/external/LFG/

James Allen:
Natural Language Understanding,
The Benjamin/Cummings Publishing Company. Inc., 2005